LED CUBE REMOTE

Final report

Instructor : Phan Duy Hùng

Team member :

Hoàng Ngọc Lộc

Nguyễn Anh Tài

Nguyễn Đức Anh

Phạm Xuân Đạt

Contents

Chapter 1 - INTRODUCTION 5
I- Introduction 5
1.1 Definition, Acronyms and Abbreviation 5
1.2 Project Introduction 5
1.2.1 Background 5
1.2.2 The Idea 7
1.2.3 Existing Projects 7
1.3 Scope of project 8
1.4 Project Overview 9
1.4.1 The Proposed System 9
1.4.2 Boundaries of the System 9
1.4.3 In-scope of capstone project 10
1.4.4 Out-scope of capstone project 10
1.4.5 Development Environment 10
1.5 Problem definition 11
1.5.1 Name of this Capstone Project 11
1.5.2 Problem Abstract 11
1.6 Reference 11
Chapter 2 - PROJECT MANAGEMENT PLAN 12
II – Project management plan: 12
2.1 Project organization: 12
2.1.1 System process model: 12
2.1.2. Roles and Responsibilities 13
2.1.3. Tools and Techniques 14
2.2 Project management plan 14
2.2.1 Tasks 14
2.2.2 Task Sheet: Assignments and Timetable 16
2.2.3 Risk management plan 17
2.3 Communication plan 18
2.3.1 Team project communication plan 18
2.3.2 Communication with supervisor 18
2.4 Codding rule: 19

Chapter 3 - System Requirement **20**
III- External Interface Requirements **20**
3.1 Use Interface 20
3.2 Hardware interface 21
3.3 System interface 22
3.4 Communications Protocol 22
IV - Functional Requirement **23**
Use case diagram 23
V – Non-functional Requirements **23**
5.1 Reliability 23
5.2 Availability 24
5.3 Maintainability 24
5.4 Usability 24
5.5 Performance 24
Chapter 4 - System Design Description **25**
VI - Design Overview **25**
VII - System Architectural Design **26**
VIII - Hardware Design **26**
8.1 Main component list: 26
8.2 Hardware details: 27
8.2.1 Arduino pro mini: 27
8.2.2 74HC574 IC 31
8.2.3 74HC138 IC: 33
8.2.4 Module HC-06: 35
8.3 Led cube design: 43
Layer led should look similar to this: 44
8.4 Circuit design: 44
8.4.1 Connect Bluetooth module hc-06 and Arduino: 44
8.4.2 Connect Arduino and multiplexer in out: 44
8.5 Box design: ratio 1:2 49
8.6 PCB design: 50
IX – Software User Interface Design **50**
X – Algorithms for LED Effects **61**

1. Default Effects 62
2. Rain 62
3. Planboing 62
4. Ripples 63
5. Axis up down random send parallel 64
6. Box shrink grow and woop woop 64
7. Filip filop 66
8. TelcStair 66
9. Linespin 67
10. Effect text – String fly 67
11. Music visualization 68

Chapter 5 - Implementation and Testing 77

XI – Implementation 77

11.1 Phase 1 77

11.2 Phase 2 79

XII - Testing 82

12.1 Test Plan 83

12.1.1 Phases test plan 83

12.1.2 Test case form 86

12.1.3 Testing tools 87

12.1.4 Test environment 87

12.2 Test Item 87

12.2.1 Phase 1 87

12.2.2 Phase 2 90

12.3 Test Report 95

12.3.1 Test report from Phase 1 95

12.3.2 Test report from Phase 2 96

Chapter 6 - User Manual 97

XIII - User Manual 97

Chapter 1 - INTRODUCTION

I- Introduction

1.1 Definition, Acronyms and Abbreviation

LCR	Led cube remote
LC	Led cube
IC	Integrated circuit

1.2 Project Introduction

1.2.1 Background

LED is the abbreviation of Light-Emitting Diode. The LED consists of a chip of 2 types of semiconducting materials P and N connecting one with another. LED lamps have a very good light performance efficiency and power saving. Appearing in the 60S of last century, LEDs were often used as practical electronic components in a number of applications such as for time display of an alarm clock or for indicator of camera's battery. However, this kind of lamps had only 3 color: green, red and yellow but not white. By 1993 a Japanese chemical company has researched and successfully made the first white LEDs and then LEDs are used more widely in reality.

Working principles of LEDs:

• Like many other types of semiconductor diodes

• Depending on the level of energy emitting high or low the wavelength light emitting different (i.e. color of the LED will vary). Energy levels (and the color of the LED) depends completely on the structure of the atomic power semiconductors materials.

• Depending on the type of LED that forward bias voltage are different. For normal LEDs, biased voltage ranges from 1.5V to 2.5V; for ultra-bright LED, the forward bias voltage may be up to 5V.

• When operating normally, the LED amperage ranges from 10mA to

Advantages of LEDs:

• Really diverse in size, color and shapes of LEDs.

• Reasonable price, costs saving

•Made from polymer materials so LEDs are durable, easy to transport without the fear of breakage.

• LEDs generates more light and have a lifespan up to 70 thousand hours of use, (if a LED lamp has 8 hours lighting a day, after 23 years it need replacing).

• Saving power up to 70 - 80% in comparison with conventional lamps.

• Insignificant thermal energy generated during operation

• Works well in low temperature conditions

• Use direct current with small voltage

• Environmental friendly because LEDs do not generate ultraviolet rays or mercury…

Applications:

LEDs are used in many fields in our life such as decorating, reading, lighting, advertising ... Especially outdoor advertising, in the toughest places for fitting or replacing lights, because we can take advantage of its many times longer lifespan than Neon lights. Moreover, they are various in colors: red, green, blue,

amber ... We will use super bright LEDs, commonly used as billboards with low power consumption, high reliability and good performance in this project.

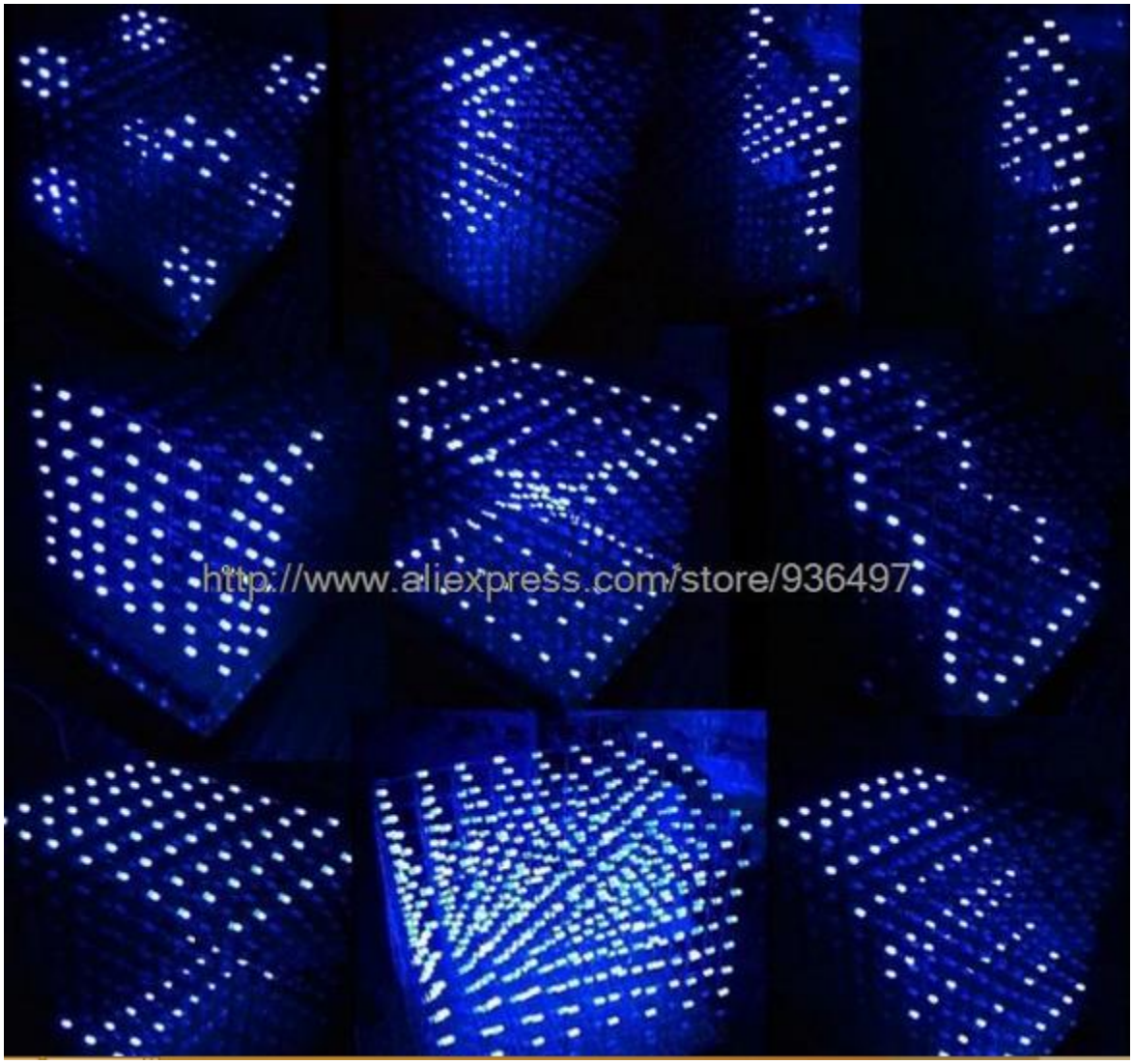
1.2.2 The Idea

Our idea is to combine multiple LEDs to create many different effects, served in the outdoor advertising or indoor decoration as a prominent furniture in the room. We use multiple LEDs connected together in a cube to create the 3D effect by turning off the lights automatically at various locations through specialized IC which is controlled by using a mini Pro Kit Arduino using 328 AT mega chip. Users will control these LEDs by using a smartphone transmitting signals to the Arduino via Bluetooth mode.

1.2.3 Existing Projects

In the current market, there are lot of similar products, however our product is exceptional and is a breakthroughs because this allows users to customize the effects of LED light box (LED cube) via a smartphone with the intuitive interface.

E.g.: http://www.aliexpress.com/item/Free-shipping-3-d-8-8-8-cubic-DIY-light-suite-USES-stm32-chip-control-technology/2028828256.html



Product Effect a very beautiful 3d effect display however LED via software loaded on the PC by a quite complicated connection is suitable for ordinary users.

1.3 Scope of project

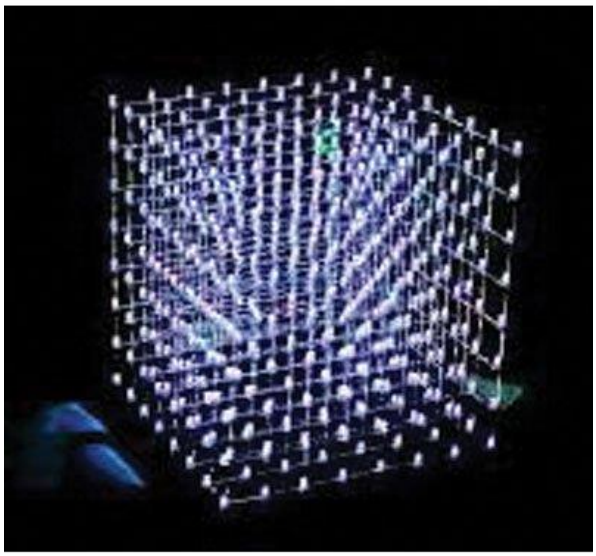
In this project we use LED cube with dimension of 8x8x8 and 512 LED lamps. The effects will be pre-loaded in the Arduino and the user will select by the software on their smartphones; the signal is transmitted to the Arduino Pro mini kit via Bluetooth module HM-11; Arduino Pro Mini processes the signal and controls

8 IC 74HC574N which controls each LED layer and IC 74HC138 to combine together these layers to create the desired effect.

1.4 Project Overview

1.4.1 The Proposed System

8*8*8 LED cube kit white by JAMECO KITPRO was sold on Amazone.com with 14.95$ using Arduino Uno to control the effect through PC software. However, this cube does not have lots of effect and only suit for experienced user. Using this cube require a PC and installation the software.



http://www.amazon.com/8x8x8-LED-Cube-Kit-White/dp/B00E4WTP06

1.4.2 Boundaries of the System

Our system was divided by 3 main parts:

- Input: user choose the effect on the smart phone.

- Process: the smart phone will process and give signal to arduino through Bluetooth. Arduino will continuing process and send command to micro processing.
- Output: the micro processing will control LED, give the effect as user wish on the 3D cube.

1.4.3 In-scope of capstone project

• When turn-on LED Cube instance running effect and changing the effect every 40 second.

• The Android app ask to turn on Bluetooth connection and connect automatic to LED Cube. This app can:

- Set the effect are running

- Change speed of effect while playing music file

1.4.4 Out-scope of capstone project

Due to limit of time and knowledge we suspect to adding more function in LED Cube and feature in the Android app in future

1.4.5 Development Environment

We using Microsoft Window as the main OS for developing both Arduino and Android application:

• C environment: using Arduino IDE for develop Arduino protocol to control LED and effect we using on LED Cube

• Java environment: using Eclipse to develop an android app which can send data to LED Cube for changing effect though Bluetooth.

1.5 Problem definition

1.5.1 Name of this Capstone Project

LED cube remote (LCR)

1.5.2 Problem Abstract

LED had been used in human life for a long time in entertaining and advertising. The combination of hundreds of LED will make an excited message or effect. A single LED can only turn on and off, light and dark. But to controll turning on and off of many LED, we could have a running message, rain dropping, bubble or any visualization human could imagine.

For example: LED was used to build a big screen, give more stage-effect for big out-door show; a huge attractive public advertisment in the street.

Our project give a solution to controll a LED cube through smart phone. By a visual simple interface, using it would be very easy, even with a general user. It could be use for home or advertisement business.

1.6 Reference

http://banlinhkien.com

http://www.aliexpress.com/

http://www.instructables.com/id/Led-Cube-8x8x8

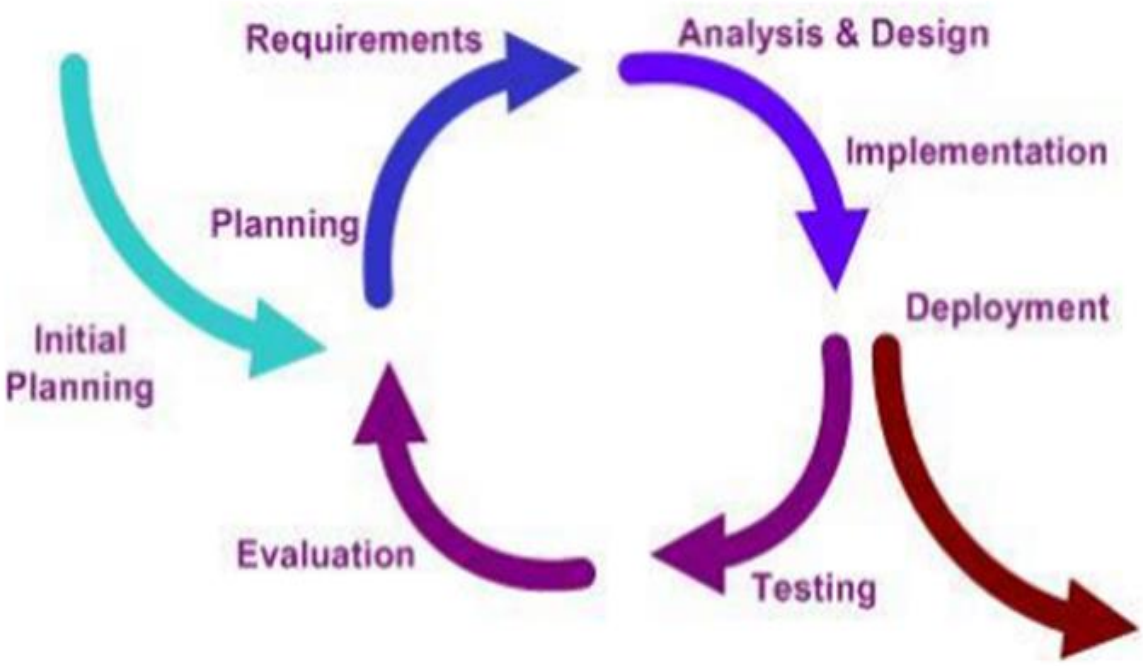
Chapter 2 - PROJECT MANAGEMENT PLAN

II – Project management plan:

2.1 Project organization:

2.1.1 System process model:

The process model used for this project is the Iterative and incremental development process model.

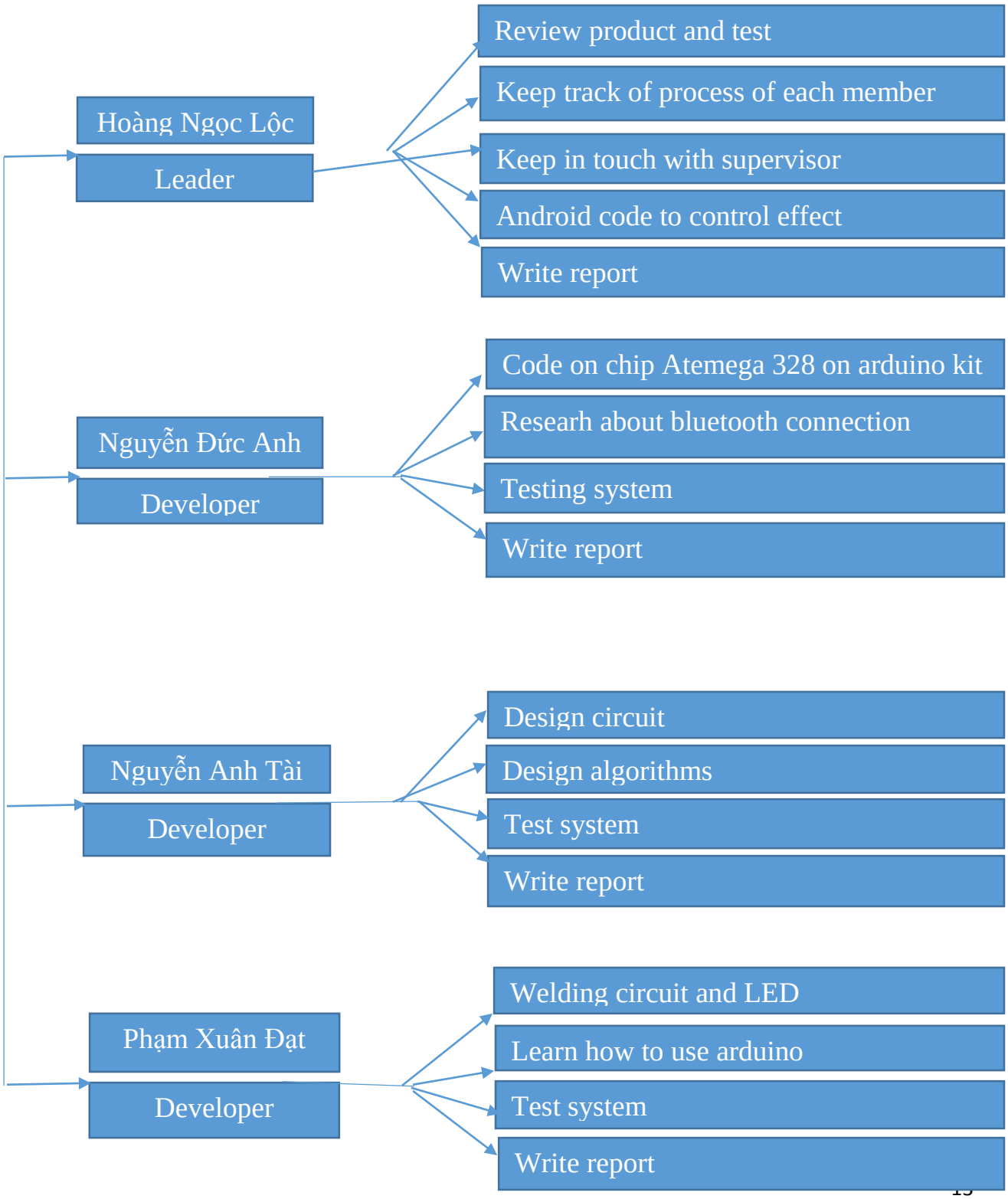


The Iterative and incremental model is a combine of Iterative model and Incremental model, it has advantages:

- Generates working product quickly and early during the product life cycle.
- Easier to test and debug during smaller iterations.
- Easier to manage risks because risky pieces are identified and handled during their iterations.

- More flexible – less costly to change scope and requirements.

2.1.2. Roles and Responsibilities



	Full Name	Phone	Email	Title
Supervisor	Phan Duy Hùng	0975597339	HungPD@fpt.edu.vn	Mr.

	Full Name	Roll number	Phone	Email	Title
Std# 1	Hoàng Ngọc Lộc	SE02006	0166639992 1	LocHNSE02006@fpt.edu.vn	PM,developer
Std# 2	Nguyễn Anh Tài	SE02273	0985539411	TaiNASE02273@fpt.edu.vn	Developer, designer, tester.
Std# 3	Nguyễn Đức Anh	SE02824	0935524011	AnhNDSE02824@fpt.edu.vn	Designer, Developer.
Std# 4	Phạm Xuân Đạt	SE02109	0169521495 5	DatPXSE02109@fpt.edu.vn	Developer, designer, tester.

2.1.3. Tools and Techniques

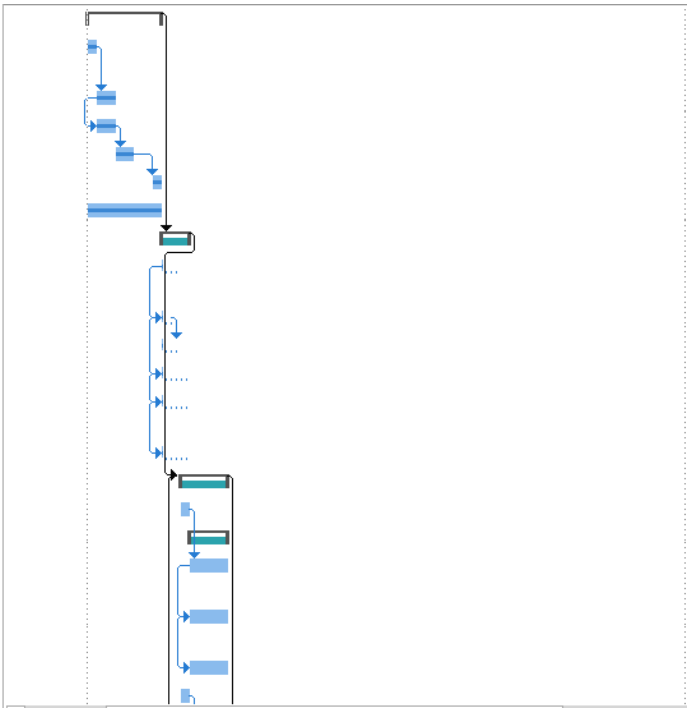
- Microsoft Office 2013 (Word, Excel, PowerPoint), Microsoft Office Visio 2013, Microsoft Project Management 2013
- VMware Workstation version 10
- Arduino version 1.0.6
- Microsoft Windows 7, 8.1
- Altium Designer version 13.3.4

2.2 Project management plan

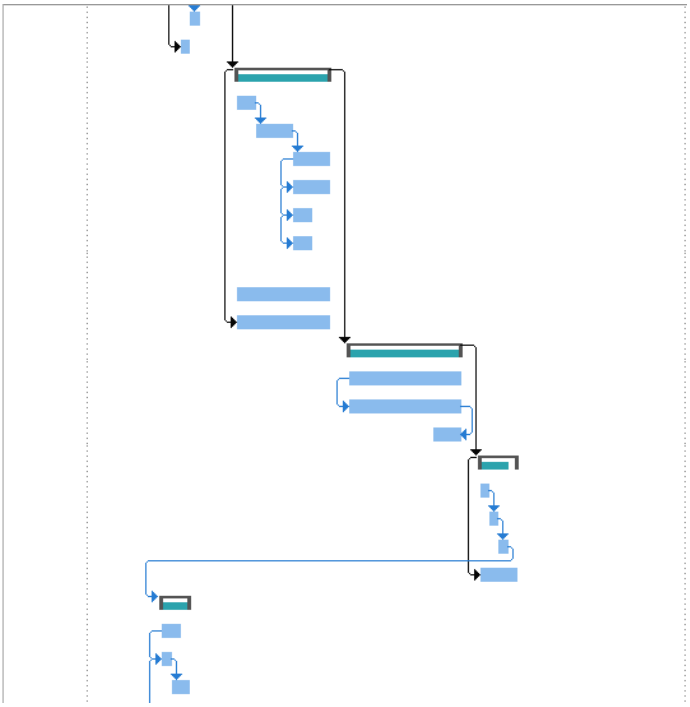
2.2.1 Tasks

Project plan:

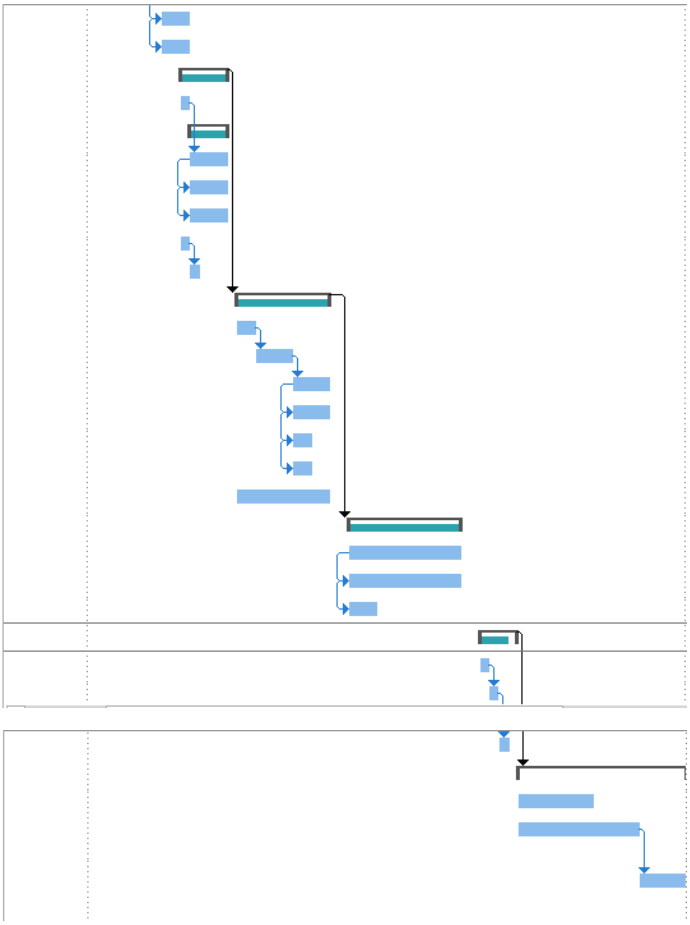
Initiation	**6 days**	**Mon 08/09/14**	**Mon 15/09/14**
Evaluation and recommendation	1 day	Mon 08/09/14	Mon 08/09/14
Develop project charte	2 days	Tue 09/09/14	Wed 10/09/14
Identify in-out scope	2 days	Tue 09/09/14	Wed 10/09/14
Identify stakeholder	2 days	Thu 11/09/14	Fri 12/09/14
Project kick-off meeti	1 day	Mon 15/09/14	Mon 15/09/14
Create report 1	6 days	Mon 08/09/14	Mon 15/09/14
Planning	**3 days**	**Tue 16/09/14**	**Thu 18/09/14**
Setup project environment	2 days	Tue 16/09/14	Wed 17/09/14
Define scope	1 day	Tue 16/09/14	Tue 16/09/14
Develop Risk plan	2 days	Tue 16/09/14	Wed 17/09/14
Develop project schec	3 days	Tue 16/09/14	Thu 18/09/14
Develop project management plan	3 days	Tue 16/09/14	Thu 18/09/14
Create report2	3 days	Tue 16/09/14	Thu 18/09/14
Prepare Requirement	**3 days**	**Thu 18/09/14**	**Mon 22/09/14**
Define Risk	1 day	Thu 18/09/14	Thu 18/09/14
Define Requirement	**2 days**	**Fri 19/09/14**	**Mon 22/09/14**
Define non-function	2 days	Fri 19/09/14	Mon 22/09/14
Define Function requirement	2 days	Fri 19/09/14	Mon 22/09/14
Define external inte	2 days	Fri 19/09/14	Mon 22/09/14
Complete SRS	1 day	Thu 18/09/14	Thu 18/09/14



Review SRS	1 day	Fri 19/09/14	Fri 19/09/14
Create report 3	1 day	Thu 18/09/14	Thu 18/09/14
Design	**8 days**	**Wed 24/09/14**	**Fri 03/10/14**
Prepare component	2 days	Wed 24/09/14	Thu 25/09/14
Architechture design	2 days	Fri 26/09/14	Mon 29/09/14
Circuit design	4 days	Tue 30/09/14	Fri 03/10/14
Protocol design	4 days	Tue 30/09/14	Fri 03/10/14
LED cube design	2 days	Tue 30/09/14	Wed 01/10/14
Software interface design	2 days	Tue 30/09/14	Wed 01/10/14
Algorthm design	8 days	Wed 24/09/14	Fri 03/10/14
Create report 4	8 days	Wed 24/09/14	Fri 03/10/14
Coding	**10 days**	**Mon 06/10/14**	**Fri 17/10/14**
Arduino code	10 days	Mon 06/10/14	Fri 17/10/14
Android code	10 days	Mon 06/10/14	Fri 17/10/14
Control code	3 days	Wed 15/10/14	Fri 17/10/14
Testing	**4 days**	**Mon 20/10/14**	**Thu 23/10/14**
System test	1 day	Mon 20/10/14	Mon 20/10/14
Intergration test	1 day	Tue 21/10/14	Tue 21/10/14
Acceptance test	1 day	Wed 22/10/14	Wed 22/10/14
Create report 5	4 days	Mon 20/10/14	Thu 23/10/14
Planning	**3 days**	**Tue 16/09/14**	**Thu 18/09/14**
Setup project environ	2 days	Tue 16/09/14	Wed 17/09/14
Define scope	1 day	Tue 16/09/14	Tue 16/09/14
Develop Risk plan	2 days	Wed 17/09/14	Thu 18/09/14



Develop project sche	3 days	Tue 16/09/14	Thu 18/09/14
Develop project mana	3 days	Tue 16/09/14	Thu 18/09/14
Prepare Requirement	**3 days**	**Thu 18/09/14**	**Mon 22/09/14**
Define Risk	1 day	Thu 18/09/14	Thu 18/09/14
Define Requirement	**2 days**	**Fri 19/09/14**	**Mon 22/09/14**
Define non-functio	2 days	Fri 19/09/14	Mon 22/09/14
Define Function re	2 days	Fri 19/09/14	Mon 22/09/14
Define external int	2 days	Fri 19/09/14	Mon 22/09/14
Complete SRS	1 day	Thu 18/09/14	Thu 18/09/14
Review SRS	1 day	Fri 19/09/14	Fri 19/09/14
Design	**8 days**	**Wed 24/09/14**	**Fri 03/10/14**
Prepare component	2 days	Wed 24/09/14	Thu 25/09/14
Architechture design	2 days	Fri 26/09/14	Mon 29/09/14
Circuit design	4 days	Tue 30/09/14	Fri 03/10/14
Protocol design	4 days	Tue 30/09/14	Fri 03/10/14
LED cube design	2 days	Tue 30/09/14	Wed 01/10/14
Software interface de	2 days	Tue 30/09/14	Wed 01/10/14
Algorthm design	8 days	Wed 24/09/14	Fri 03/10/14
Coding	**10 days**	**Mon 06/10/14**	**Fri 17/10/14**
Arduino code	10 days	Mon 06/10/14	Fri 17/10/14
Android code	10 days	Mon 06/10/14	Fri 17/10/14
Control code	3 days	Mon 06/10/14	Wed 08/10/14
Testing	**4 days**	**Mon 20/10/14**	**Thu 23/10/14**
System test	1 day	Mon 20/10/14	Mon 20/10/14
Intergration test	1 day	Tue 21/10/14	Tue 21/10/14
Acceptance test	1 day	Wed 22/10/14	Wed 22/10/14
Closing	**12 days**	**Fri 24/10/14**	**Mon 10/11/14**
Create report 6	6 days	Fri 24/10/14	Fri 31/10/14
complete project document	9 days	Fri 24/10/14	Wed 05/11/14
Prepare to final project presentation	3 days	Thu 06/11/14	Mon 10/11/14



2.2.2 Task Sheet: Assignments and Timetable

No.	Name	Mission
1	LocHN	Connect part of project, part of task, creat plan for every week and report result. Android coder, Arduino coder.
2	DatPX	Welding circuit, Arduino coder, create Requirement
3	AnhND	Program control function for the LED cube effect via Arduino. Android and Arduino Bluetooth connection.

4	TaiNa	Research function for control LED cube, Android coder, Tester.

2.2.3 Risk management plan

ID	Risk description	Probability	Effect	Status	Action
1	Project manager lack of management skill.	Very High	Serious		Improve management skill.
2	Don't understand about single circuit module and design hardware.	Very High	Serious		Search about principle on internet, buy some circuits' module and redesign mainboard.
3	Cannot control member to follow schedule.	High	Serious		Keep in touch with all member, tracking task by report.
4	Break component while processing.	High	Tolerable		Buy spare components, check carefully before testing
5	Team member may get accident, sick, family problem.	Moderate	Tolerable		Estimate buffer time in project plan and require all team member to inform team 1 week.

2.3 Communication plan

2.3.1 Team project communication plan

Team meeting online 3 times per week at 7:00pm to 9:30 pm Tuesday, Thursday and Saturday, meeting offline at 2:00 to 5:00 pm Sunday afternoon at DatPX's room

Implement Phase: All members implement all task that PM and technical leader assigned on TFS (Team Foundation Server). Every week, on Tuesday, there will be have a meeting when each team member report to the whole team what he did the previous week, then all member plan the schedule for upcoming week, and solve the issues together.

Communication Channels:

- Email: Gmail.
- Instant message: Skype.
- Communication forum: Facebook private group.
- Phone: in urgent emergence, phone is a good ways to share the idea.
- Face to face meetings.

2.3.2 Communication with supervisor

Team meets the supervisor every week at Sunday follow time table in Fap.fpt.edu.vn to get advices for issues, guide for writing and correcting reports and plan for the next week. When having troubles in project, project manager collects ideas from members and send email or have a meeting with supervisor to get instructions or advices. By the report, supervisor will keep in touch with project

2.4 Codding rule:

Android code will follow java coding rule:

http://www.oracle.com/technetwork/java/codeconvtoc-136057.html

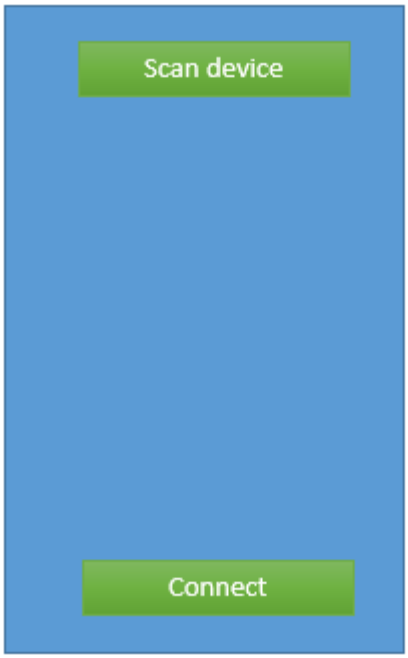
Arduino code will follow C coding rule:

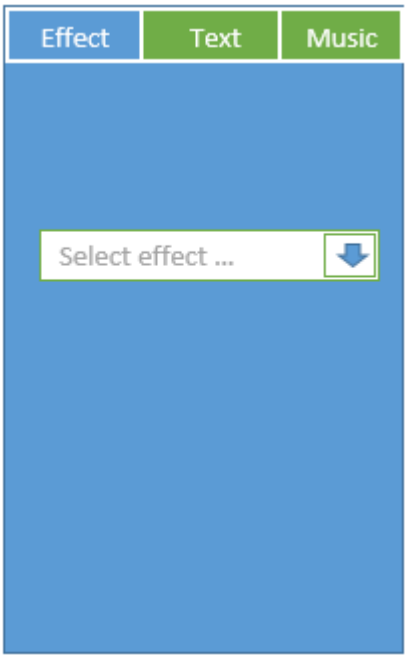
http://users.ece.cmu.edu/~eno/coding/CCodingStandard.html

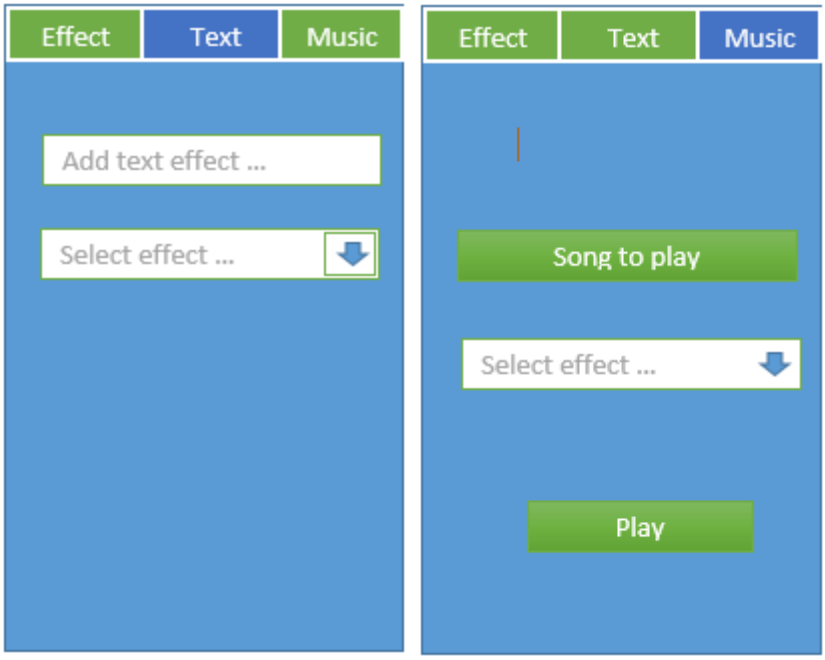
Chapter 3 - System Requirement

III- External Interface Requirements

3.1 Use Interface



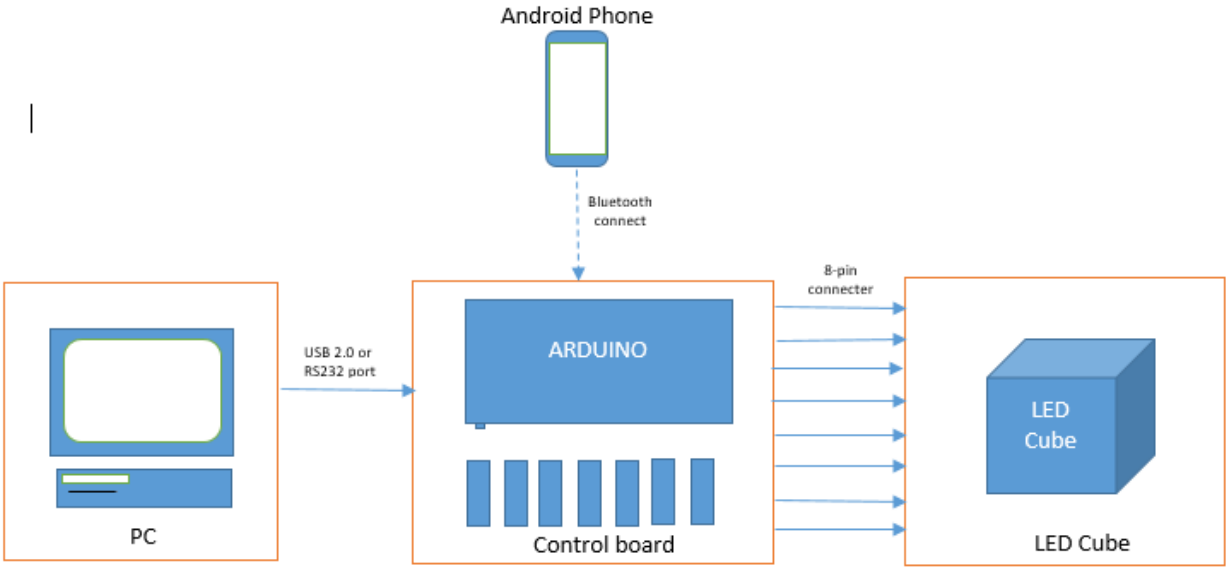




3.2 Hardware interface

- PC must have RS232 port or usb port to connect module, Performance at least: 40 GB free, 1 GB ram, core i3 1.5 GHz.
- Arduino kit has com port or mini usb port or can use USB serial adapter to convert to USB port. Have enough pin to connect 8 microchip, use At mega 328 microprocessor.

3.3 System interface

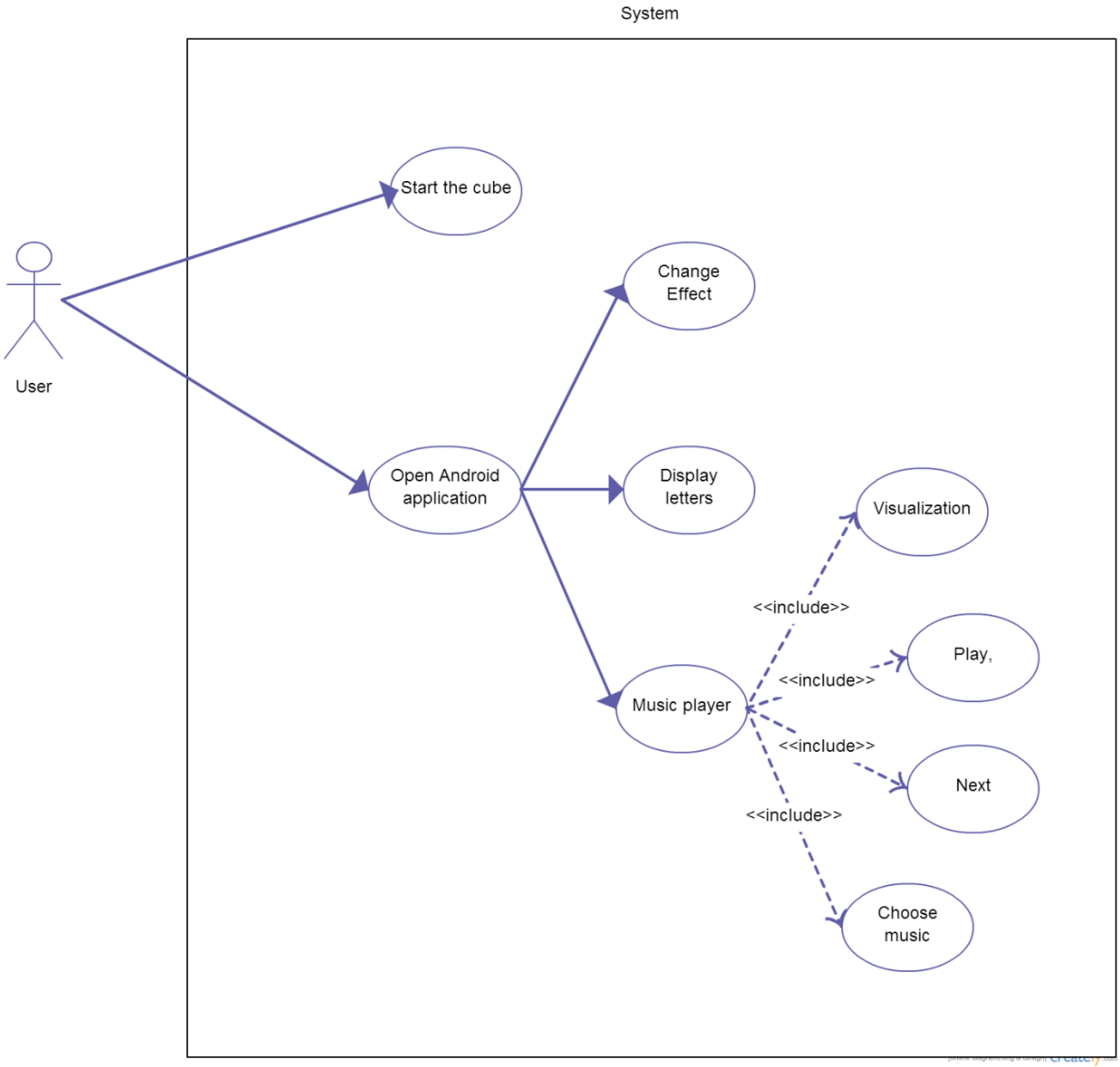


3.4 Communications Protocol

- Use RS232 or USB protocol to connect control board to PC.
- Use Bluetooth 4.0 protocol to connect android Phone to control board.
- Use 8-pin connecter to connect from control board to LED Cube.

IV - Functional Requirement

Use case diagram



Use case diagram of LED Cube

V – Non-functional Requirements

5.1 Reliability

- The system activities should be reliable and correct to the user's choice.

- The connection between smartphone and the hardware must to ensure stable operation, when they were connected, the unintended disconnection must not occur.

5.2 Availability

- The system must be ready to operate at any time when it's started by users.
- The system’s able to ensure the stable operation continuously for at least 12 hours.

5.3 Maintainability

- Modularized design.
- Clearly following coding convention.

5.4 Usability

- English user interface design, easy to understand, reasonable.
- Direct instruction in software.

5.5 Performance

- The system should run fast , when users press button “Start “ to run system, booting Led effect must be display no later than 2 seconds
- When users chose effects on mobile then they sent, effect must display no later than 1 sec

Chapter 4 - System Design Description

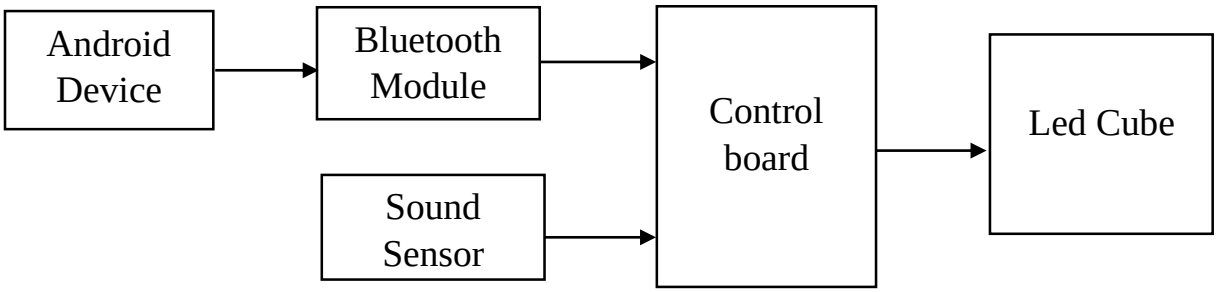
VI - Design Overview

Remote LED Cube (RLC) is a project using Android device to control how a LED Cube works. Basically we using an Android application connect to an 8x8x8 led cube thought Bluetooth connection. The Android application can choose how the led effect changes, can send text and choose how the text appear in led cube, and the application has a music player which can using led cube as a visualization.

This document describes the technical and UI design of RLC. It contain the architecture design, detail hardware design and user interface of Android application. It also include the algorithms we use to make effect and visualization for 8x8x8 led cube:

- The Architecture design describes the overall architecture of the system, and how each component works in working environment.
- The effect algorithms describe each effect we use on LED Cube and how it appear.
- Hardware design describes the hardware use in RLC, control broad design, led cube design and box design which contain all the components.
- UI design describes the layout of application and some alternative design of the user interface.

VII - System Architectural Design



VIII - Hardware Design

8.1 Main component list:

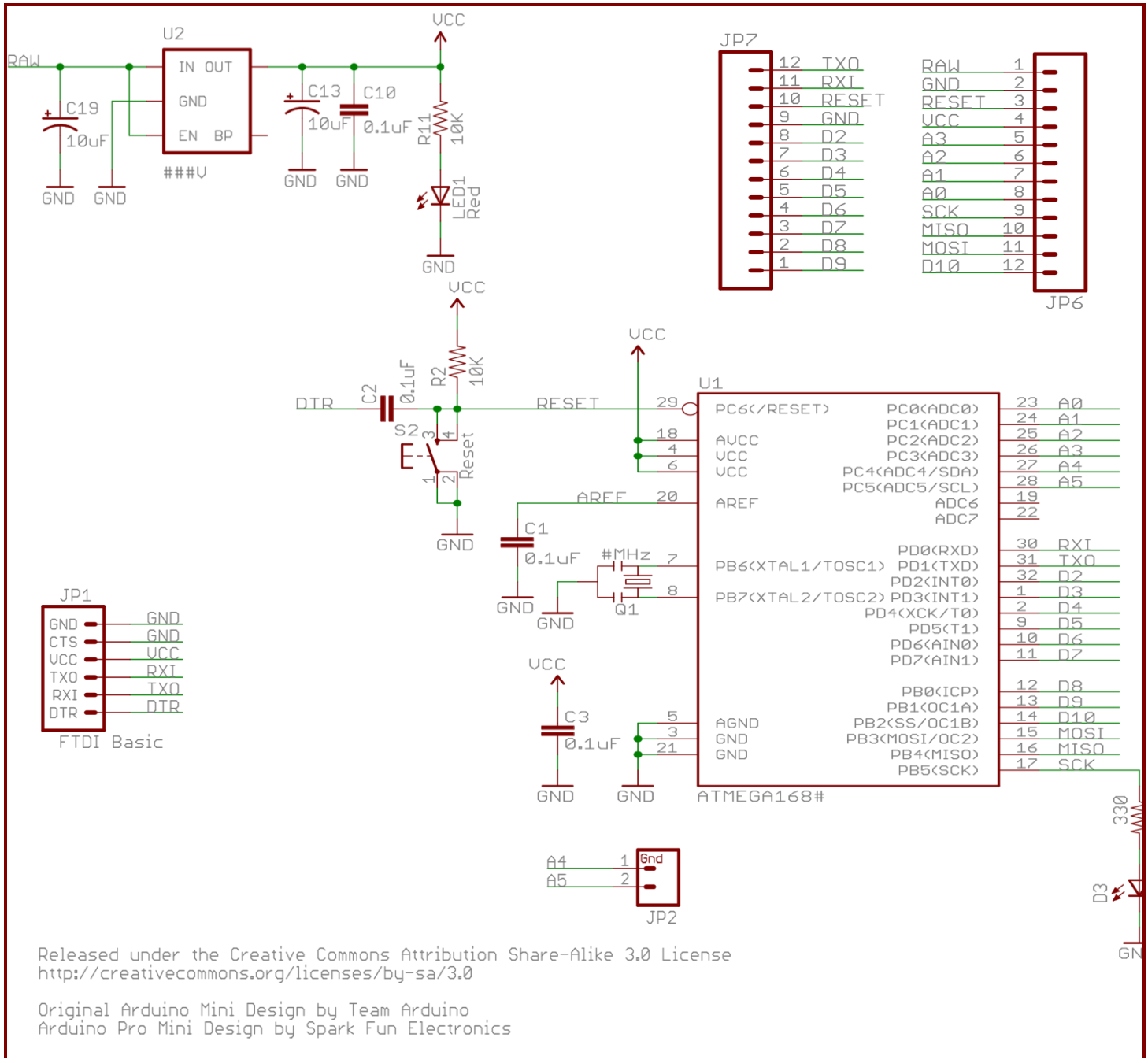
Name	Quantity	Value (if available)
Led	515	
Resistor	75	100 ohm
Resistor	16	1000 ohm
Large prototype PCB	1	
Arduino pro mini	1	
74HC574 IC	8	
74HC138 IC	1	
PN2222	16	
Ceramic capacitor	2	22pF
Ceramic capacitor	16	0.1uF
Electrolytic capacitor	3	1000 uF
Electrolytic capacitor	3	10 uF
Electrolytic capacitor	1	100 uF
Pin IC socket 20	8	
Pin IC socket 16	2	

Piece of wood	1	
Module HC-06	1	
Power suply	1	5V

8.2 Hardware details:

8.2.1 Arduino pro mini:

a. Schematic:



b. Overview:

The Arduino Pro Mini is a microcontroller board based on the ATmega168. It has 14 digital input/output pins (of which 6 can be used as PWM – pulse width modulation - outputs), 8 analog inputs, an on-board resonator, a reset button, and holes for mounting pin headers. A six pin header can be connected to an FTDI cable or Spark fun breakout board to provide USB power and communication to the board.

c. Power:

The Arduino Pro Mini can be powered with an FTDI cable or breakout board connected to its six pin header, or with a regulated 3.3V or 5V supply (depending on the model) on the VCC pin. There is a voltage regulator on board so it can accept voltage up to 12VDC. If you're supplying unregulated power to the board, be sure to connect to the "RAW" pin on not VCC. The power pins are as follows:

- RAW: For supplying a raw voltage to the board.
- VCC: The regulated 3.3 or 5 volt supply.
- GND: Ground pins.

d. Memory:

The ATmega168 has 16 KB of flash memory for storing code (of which 2 KB is used for the boot loader). It has 1 KB of SRAM and 512 bytes of EEPROM (which can be read and written with the EEPROM library).

e. Input and output:

Each of the 14 digital pins on the Pro Mini can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 3.3 or 5 volts (depending on the model). Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the TX-0 and RX-1 pins of the six pin header.
- External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function.
- SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language.
- LED: 13. There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

The Pro Mini has 8 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). Four of them are on the headers on the edge of the board; two (inputs 4 and 5) on holes in the interior of the board. The analog inputs measure from ground to VCC. Additionally, some pins have specialized functionality:

- I^2C: A4 (SDA) and A5 (SCL). Support I^2C (TWI) communication using the Wire library.

There is another pin on the board:

- Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

See also the mapping between Arduino pins and ATmega168 ports.

f. Communication:

The Arduino Pro Mini has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega168 provides UART TTL serial communication, which is available on digital pins 0 (RX) and 1 (TX). The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board via a USB connection.

A Software Serial library allows for serial communication on any of the Pro Mini's digital pins.

The ATmega168 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus.

g. Automatic (software) reset:

Rather than requiring a physical press of the reset button before an upload, the Arduino Pro Mini is designed in a way that allows it to be reset by software running on a connected computer. One of the pins on the six-pin header is connected to the reset line of the ATmega168 via a 100 nanofarad capacitor. This pin connects to one of the hardware flow control lines of the USB-to-serial convertor connected to the header: RTS when using an FTDI cable, DTR when using the Spark fun breakout board. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the

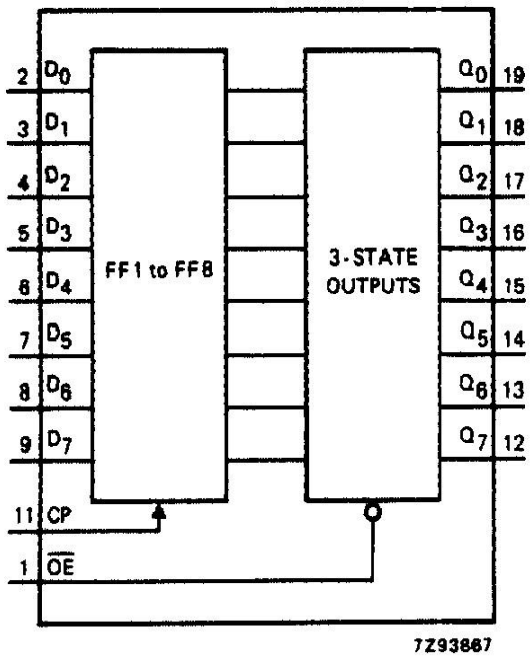
Arduino environment. This means that the boot loader can have a shorter timeout, as the lowering of the reset line can be well-coordinated with the start of the upload.

This setup has other implications. When the Pro Mini is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the boot loader is running on the Pro. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

8.2.2 74HC574 IC

The 74HC/HCT574 are octal D-type flip-flops featuring separate D-type inputs for each flip-flop and non-inverting 3-state outputs for bus oriented applications. A clock (CP) and an output enable (OE) input are common to all flip-flops. The 8 flip-flops will store the state of their individual D-inputs that meet the set-up and hold time requirements on the LOW-to-HIGH CP transition. When OE is LOW, the contents of the 8 flip-flops are available at the outputs. When OE is HIGH, the outputs go to the high impedance OFF-state. Operation of the OE input does not affect the state of the flip-flops.

a. Table function:



Functional diagram

OPERATING MODES	INPUTS			INTERNAL FLIP-FLOPS	OUTPUTS
	$\overline{OE}$	CP	D_n		Q_0 to Q_7
load and read register	L	↑	l	L	L
	L	↑	h	H	H
load register and disable outputs	H	↑	l	L	Z
	H	↑	h	H	Z

NOTE:

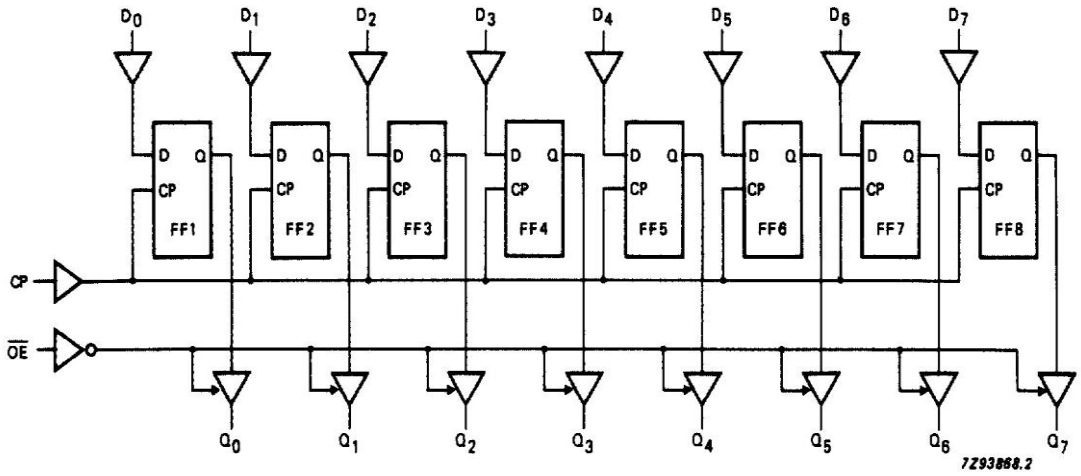
H = HIGH voltage level h = HIGH voltage level one set-up time prior to the LOW-to-HIGH CP transition L = LOW voltage level

l = LOW voltage level on set-up time prior to the LOW-to-HIGH CP transition

Z = HIGH impedance OFF-state

↑ = LOW-to-HIGH clock transition

b. Logic diagram:



Logic diagram

8.2.3 74HC138 IC:

The 74HC/HCT138 decoders accept three binary weighted address inputs (A0, A1, A2) and when enabled, provide 8 mutually exclusive active LOW outputs (Y0to Y7). The “138” features three enable inputs: two active LOW (E1andE2) and one active HIGH (E3). Every output will be HIGH unlessE1andE2are LOW and E3is HIGH. This multiple enable function allows easy parallel expansion of the “138” to a 1-of-32 (5 lines to 32 lines) decoder with just four “138” ICs and one inverter. The ”138” can be used as an eight output demultiplexer by using one of the active LOW enable inputs as the data input and the remaining enable inputs as strobes. Unused enable inputs must be permanently tied to their appropriate active HIGH or LOW state.

a. Table function:

Inputs						Outputs							
E 1	**E 2**	**E 3**	**A 0**	**A 1**	**A 2**	**Y 0**	**Y 1**	**Y 2**	**Y 3**	**Y 4**	**Y 5**	**Y 6**	**Y 7**
H	X	X	X	X	X	H	H	H	H	H	H	H	H
X	H	X	X	X	X	H	H	H	H	H	H	H	H
X	X	L	X	X	X	H	H	H	H	H	H	H	H
L	L	H	L	L	L	L	H	H	H	H	H	H	H
L	L	H	H	L	L	H	L	H	H	H	H	H	H
L	L	H	L	H	L	H	H	L	H	H	H	H	H
L	L	H	H	H	L	H	H	H	L	H	H	H	H
L	L	H	L	L	H	H	H	H	H	L	H	H	H
L	L	H	H	L	H	H	H	H	H	H	L	H	H
L	L	H	L	H	H	H	H	H	H	H	H	L	H
L	L	H	H	H	H	H	H	H	H	H	H	H	L

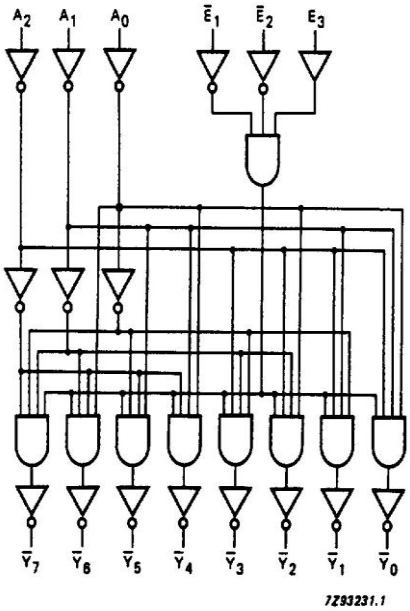
Note:

L = LOW voltage level

X = don’t care

H = HIGH voltage level

b. Logic diagram



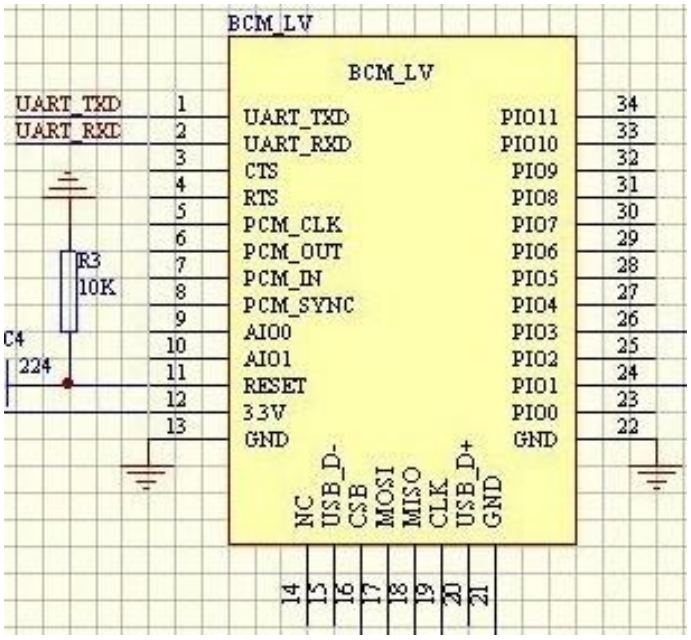
Logic diagram

8.2.4 Module HC-06:

Master and slave mode can't be switched. Master role: have paired memory to remember last slave device and only make pair with that device unless KEY (PIN26) is triggered by high level. The default connected PIN26 is low level. Pairing: Master device search and make pair with the slave device automatically. Typical method: On some specific conditions, master and slave device can make pair with each other automatically.

Multi-device communication: There is only point to point communication for modules, but the adapter can communicate with multi-modules.AT Mode: Before paired, it is at the AT mode. After paired it's at transparent communication. During the communication mode, the module can't enter to the AT mode.Default communication baud rate: 9600.

a. PINs description



PIN configuration

The PINs at this block diagram is as same as the physical one.

PIN Name	PIN #	Pad type	Description	Note
GND	13 21 22	VSS	Ground pot	
1V8	14	VDD	Integrated 1.8V (+) supply with On-chip linear regulator output within 1.7-1.9V	
VCC	12	3.3V		
AIO0	9	Bi-Directional	Programmable input/output line	
AIO1	10	Bi-Directional	Programmable input/output line	

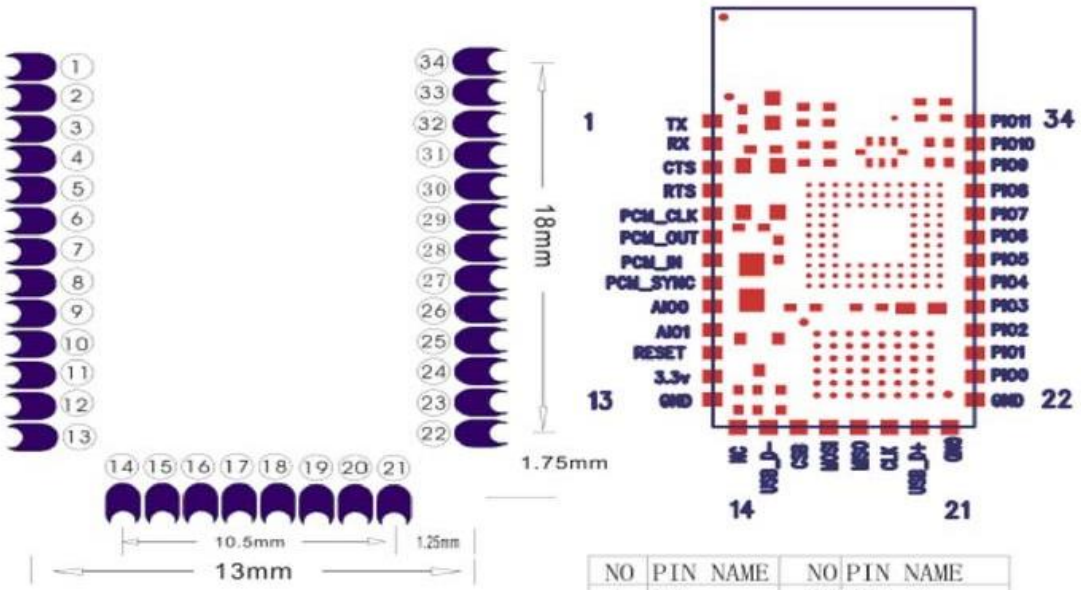
PIO0	23	Bi-Directional RX EN	Programmable input/output line, control output for LNA(if fitted)	

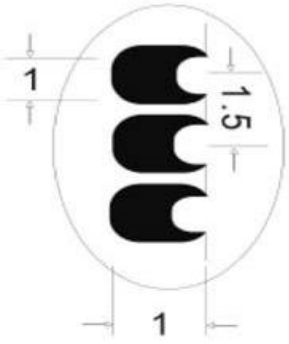
PIO1	24	Bi-Directional TX EN	Programmable input/output line, control output for PA(if fitted)	
PIO2	25	Bi-Directional	Programmable input/output line	
PIO3	26	Bi-Directional	Programmable input/output line	
PIO4	27	Bi-Directional	Programmable input/output line	
PIO5	28	Bi-Directional	Programmable input/output line	
PIO6	29	Bi-Directional	Programmable input/output line	CLK_REQ
PIO7	30	Bi-Directional	Programmable input/output line	CLK_OUT
PIO8	31	Bi-Directional	Programmable input/output line	
PIO9	32	Bi-Directional	Programmable input/output line	
PIO10	33	Bi-Directional	Programmable input/output line	
PIO11	34	Bi-Directional	Programmable input/output line	
RESETB	11	CMOS Input with weak intimal pull-down		
UART_RTS	4	CMOS output, tri-stable with weak internal pull-up	UART request to send, active low	

UART_CTS	3	CMOS input with weak internal pull-down	UART clear to send, active low	
UART_RX	2	CMOS input with weak internal pull-down	UART Data input	
UART_TX	1	CMOS output, Tri-stable with weak internal pull-up	UART Data output	
SPI_MOSI	17	CMOS input with weak internal pull-down	Serial peripheral interface data input	
SPI_CSB	16	CMOS input with weak internal	Chip select for serial peripheral interface, active low	
		pull-up		
SPI_CLK	19	CMOS input with weak internal pull-down	Serial peripheral interface clock	

SPI_MISO	18	CMOS input with weak internal pull-down	Serial peripheral interface data Output	
USB_-	15	Bi-Directional		
USB_+	20	Bi-Directional		
1.8V	14		1.8V external power supply input	Default : 1.8V internal powe r supply.
PCM_CLK	5	Bi-Directional		
PCM_OUT	6	CMOS output		
PCM_IN	7	CMOS Input		
PCM_SYNC	8	Bi-Directional		

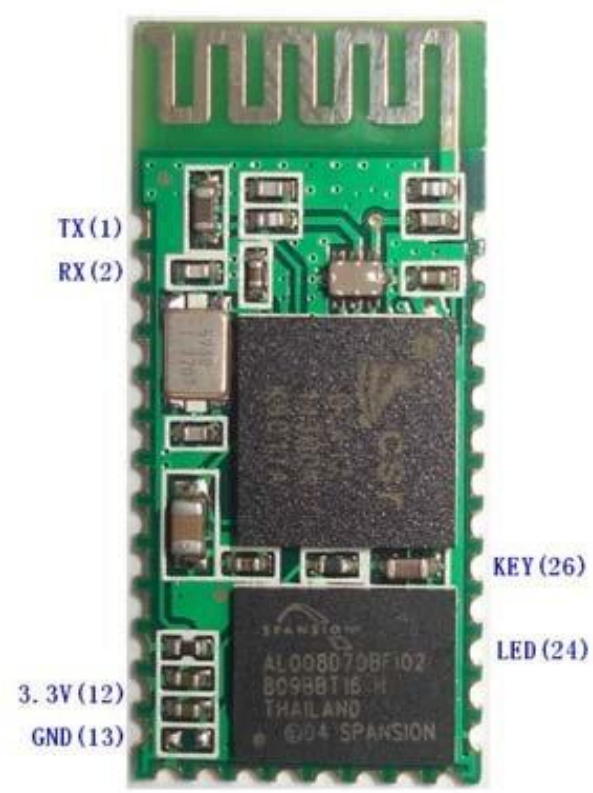
HC 06 main pins





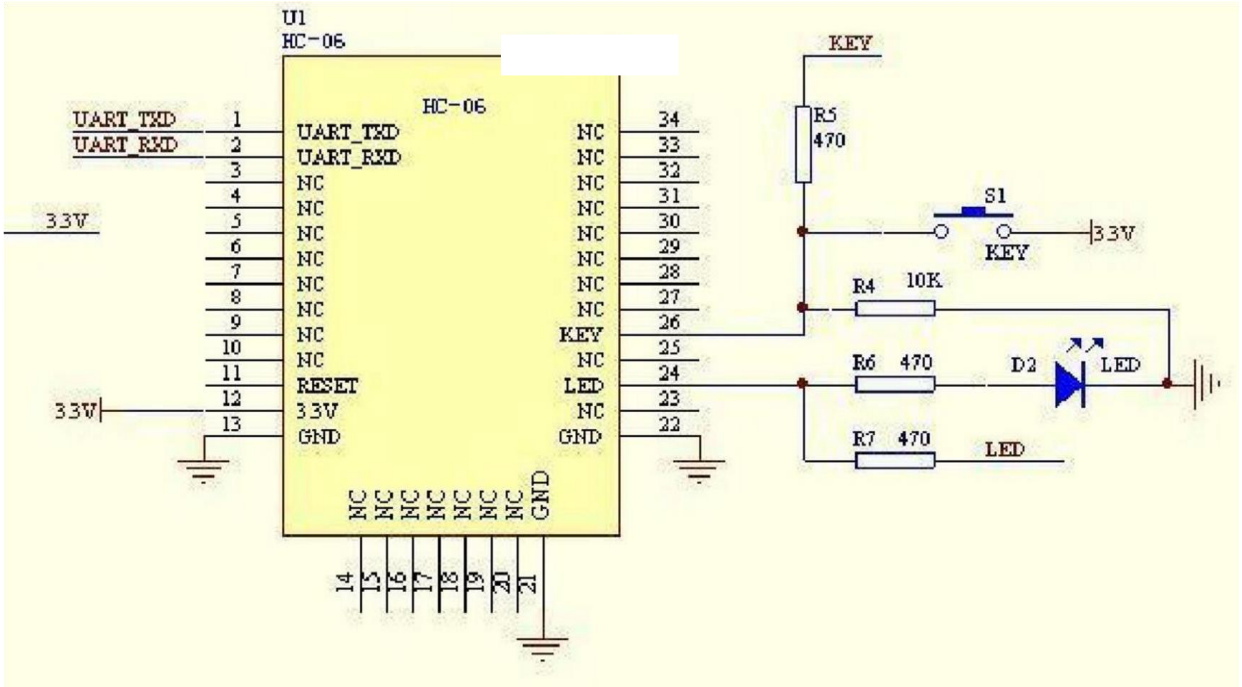
PCB Layout 请参考实物

NO	PIN NAME	NO	PIN NAME
1	TX	20	USB D+
2	RX	21	GND
3	CTS	22	GND
4	RTS	23	PIO0
5	PCM CLK	24	PIO1
6	PCM OUT	25	PIO2
7	PCM IN	26	PIO3
8	PCM SYNC	27	PIO4
9	AIO0	28	PIO5
10	AIO1	29	PIO6
11	RESET	30	PIO7
12	3.3V	31	PIO8
13	GND	32	PIO9
14	NC	33	PIO10
15	USB D-	34	PIO11
16	CSB		
17	MOSI		
18	MISO		
19	CLK		



Package size information

b. Connect the module to 3.3V serial port of MCU – microprocessor control unit



In principle, HC-06 can work when UART_TXD, UART_RXD, VCC and GND are connected. However, for better testing results, connecting LED and KEY are recommended (when testing the master).

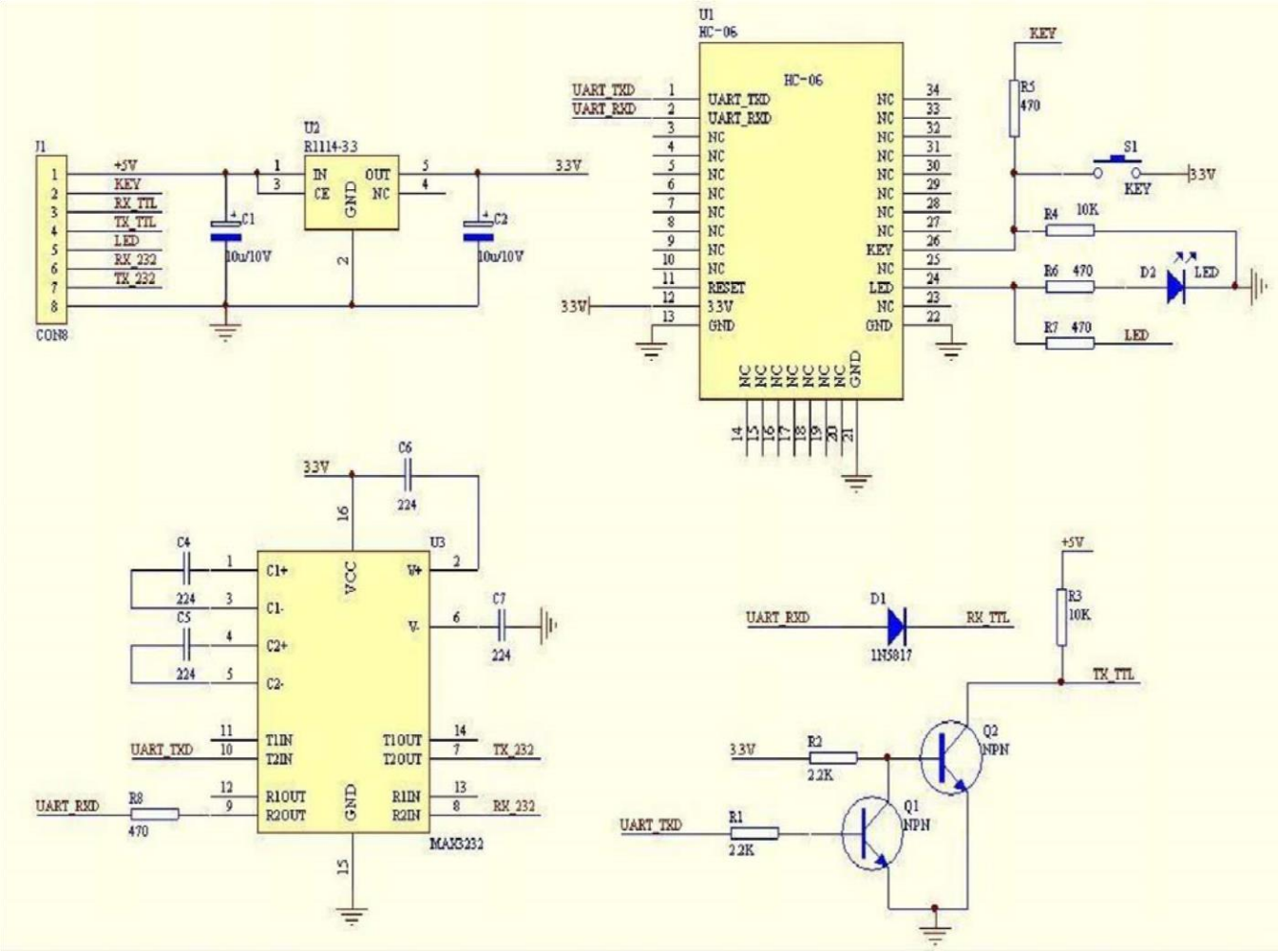
Where, the 3.3V TXD of MCU connects to HC-06's UART_RXD, the 3.3V RXD of MCU connects to HC-06's UART_TXD, and 3.3V power and GND should be connected. Then the minimum system is finished.

Note that, the PIN2:UART_RXD of Bluetooth module has no pull-up resistor. If the MCU TXD doesn't have pull-up function, then user should add a pull-up resistor to the UART_RXD. It may be easy to be ignored.

If there are two MCU which connect to master and slave device respectively, then before paired (LED will flicker) user can send AT commands by serial port when the system is power on.

If the LED is constant lighting, it indicates the pairing is finished. The two MCUs can communicate with each other by serial port. User can think there is a serial port line between two MCUs.

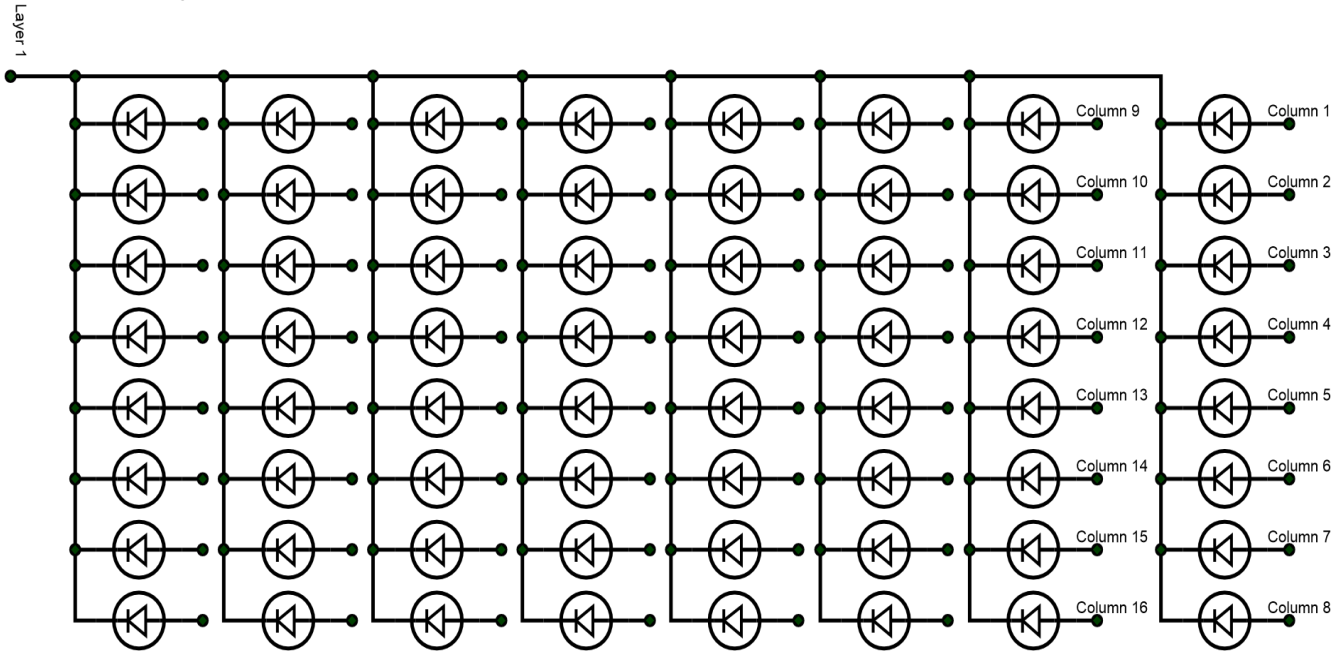
c. Connect the module to 5V serial port of MCU



8.3 Led cube design:

This led is made up of columns and layers. The cathode legs of every led in a layer are soldered together. All the anode legs in one column are soldered together. Therefore we will have 64 columns (anode) and 8 layers (cathode). Each of them are connected to the controller board with a separate wire and can be controlled individually. The led cube will be place in a piece of wood for easier wiring.

Layer led should look similar to this:



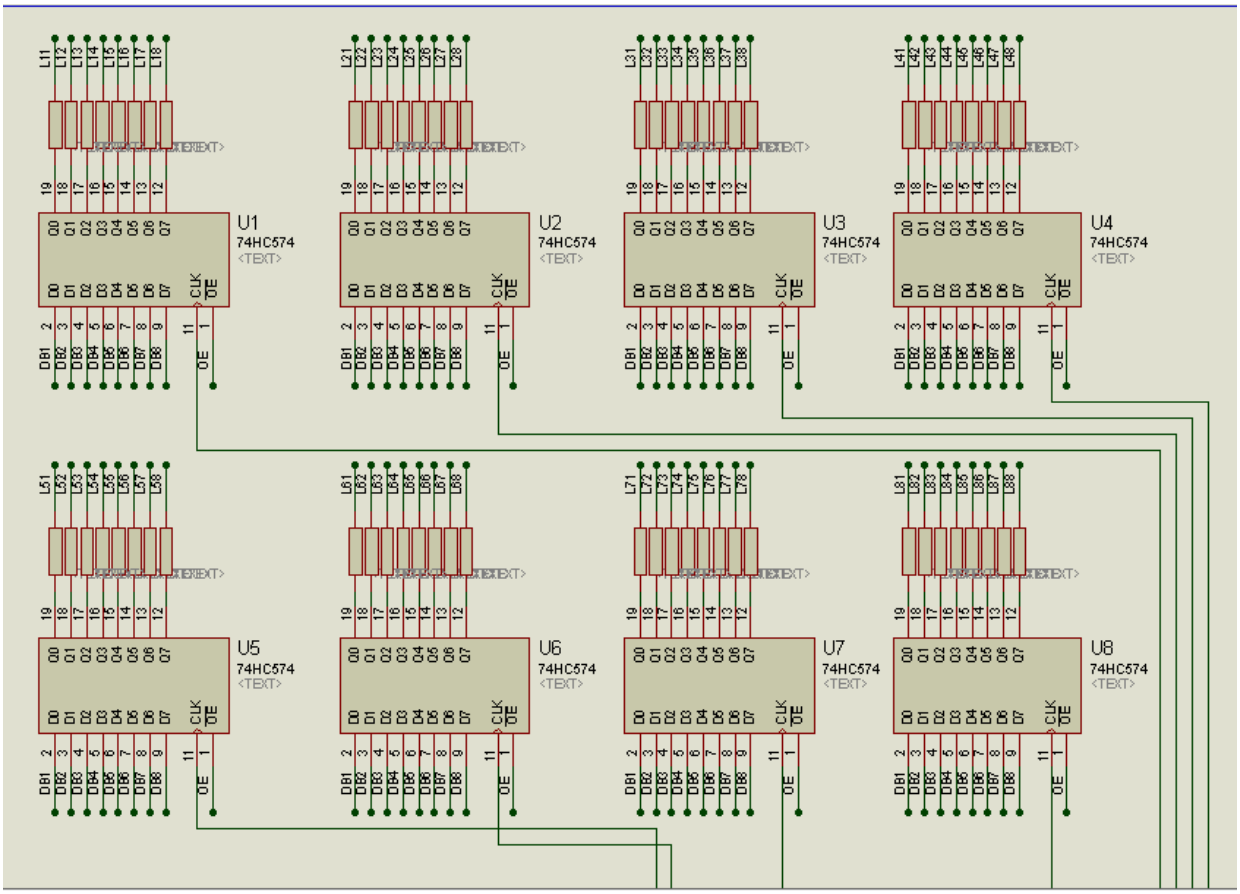
8.4 Circuit design:

8.4.1 Connect Bluetooth module hc-06 and Arduino:

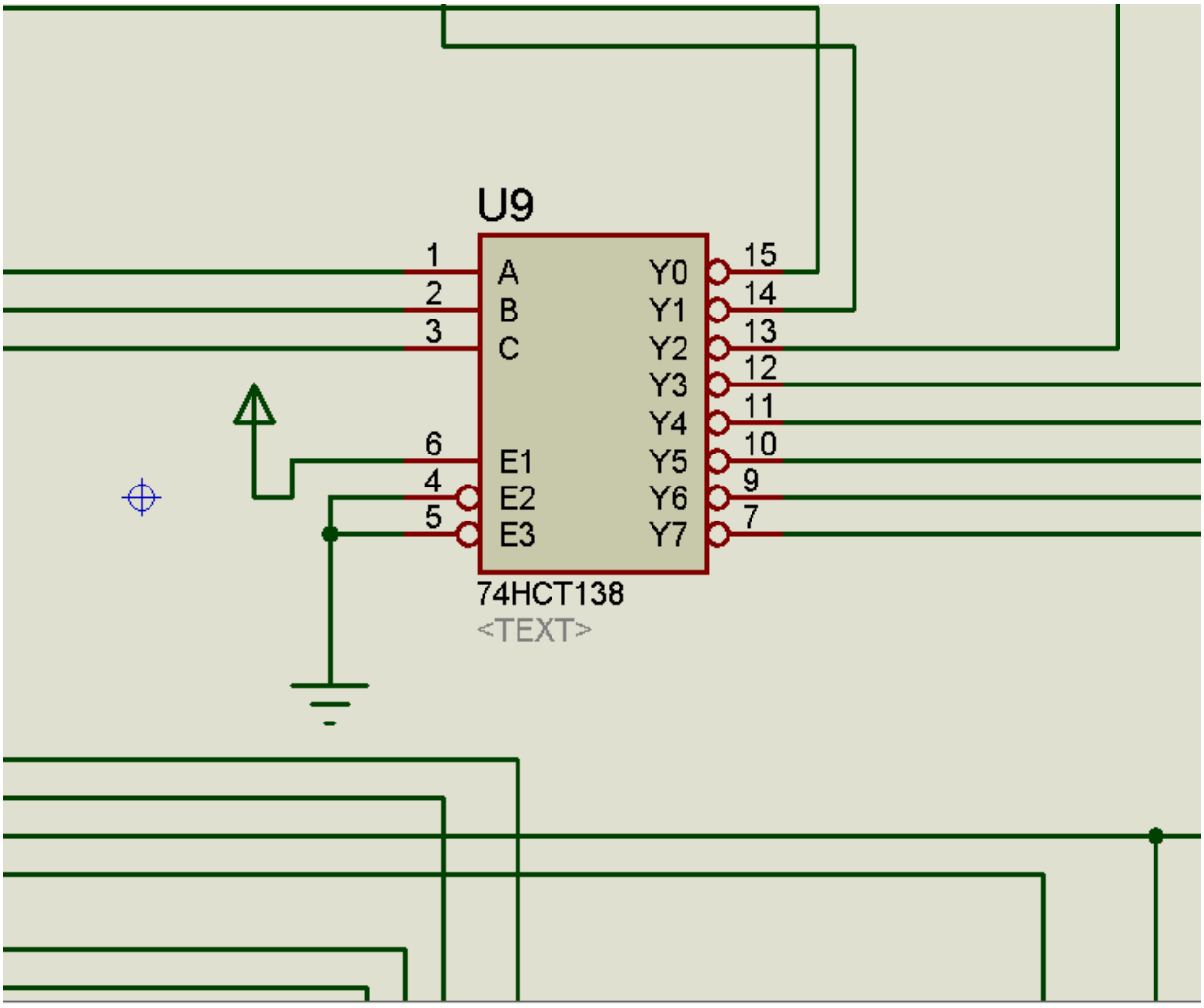
5V	→	5V
GND	→	GND
TX	→	TX
RX	→	RX

8.4.2 Connect Arduino and multiplexer in out:

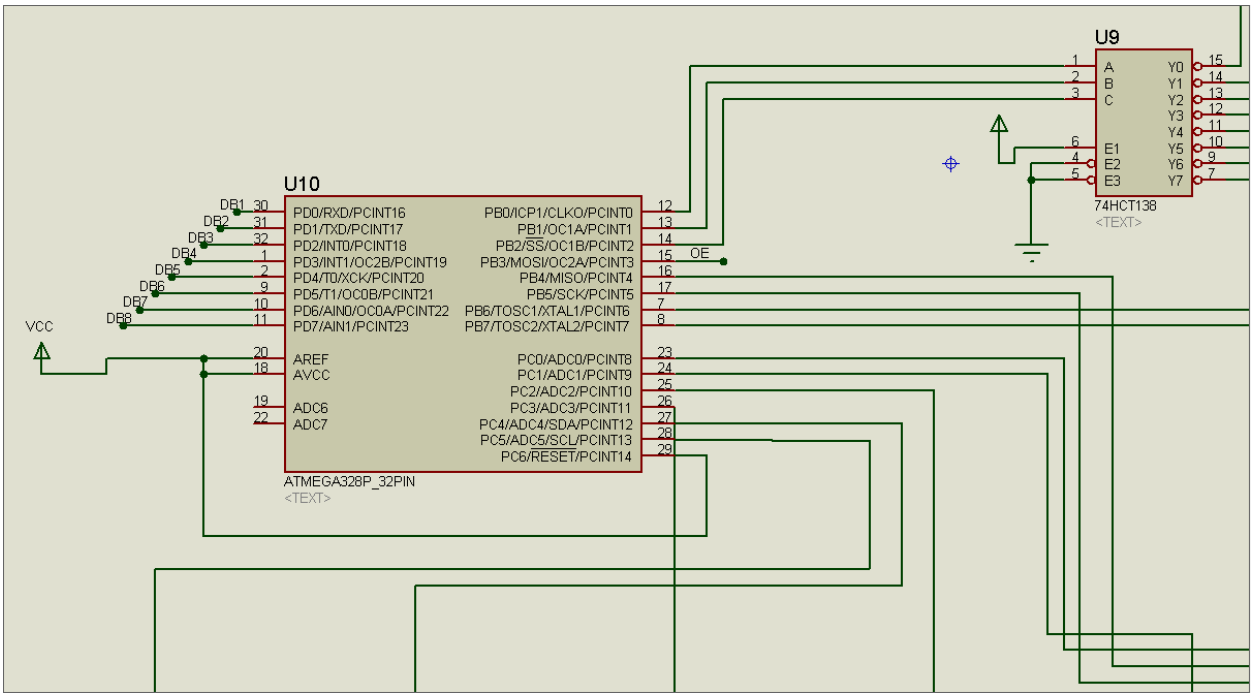
This circuit simulate how to connect to led cube through in-out port using the core of Arduino pro mini – ATMEGA chip with a set-up crystal.



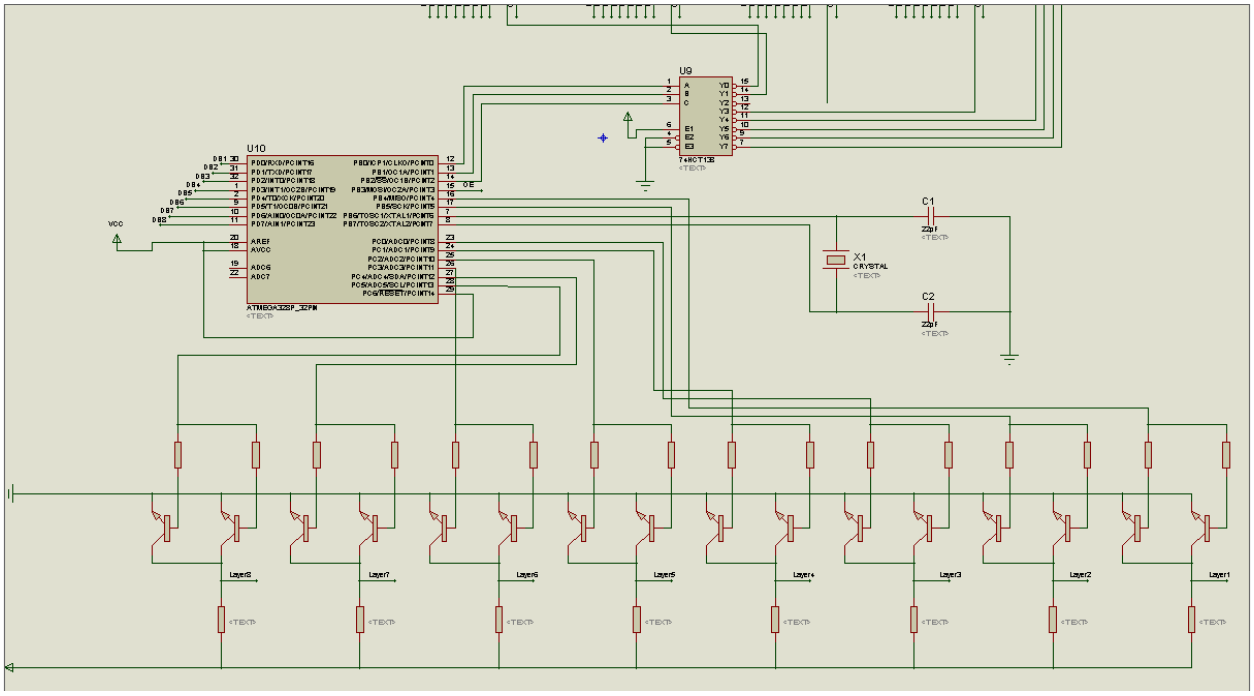
In 1 layer led, each row of 8 leds will be controlled by IC 74HC574



8 IC 74HC574 will be controlled by IC 74 HCT138

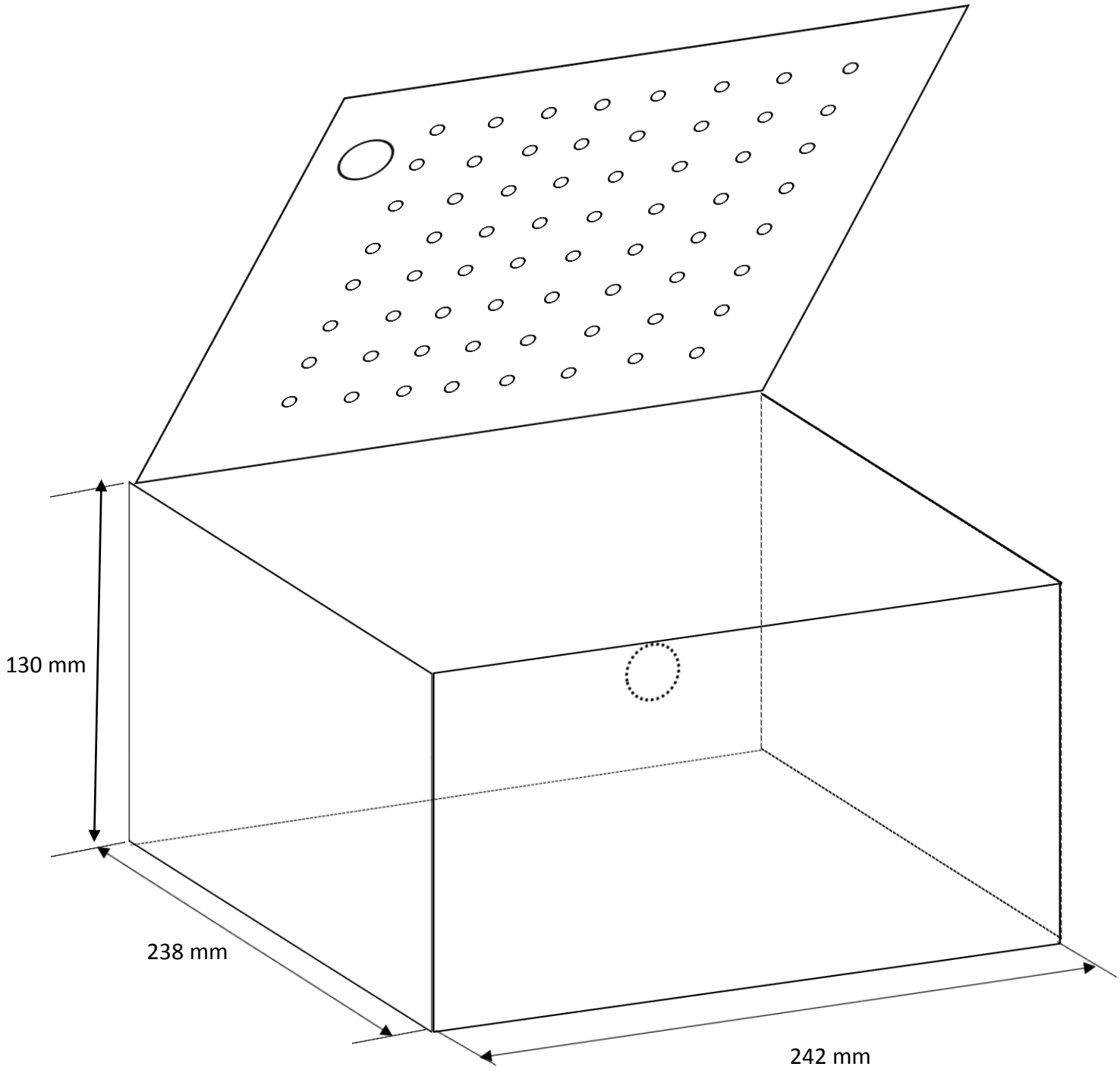


All data bus for each row of 8 leds will be connected to ATMEGA chip as well as a pin to enable output and 3 pins control IC 74HCT138

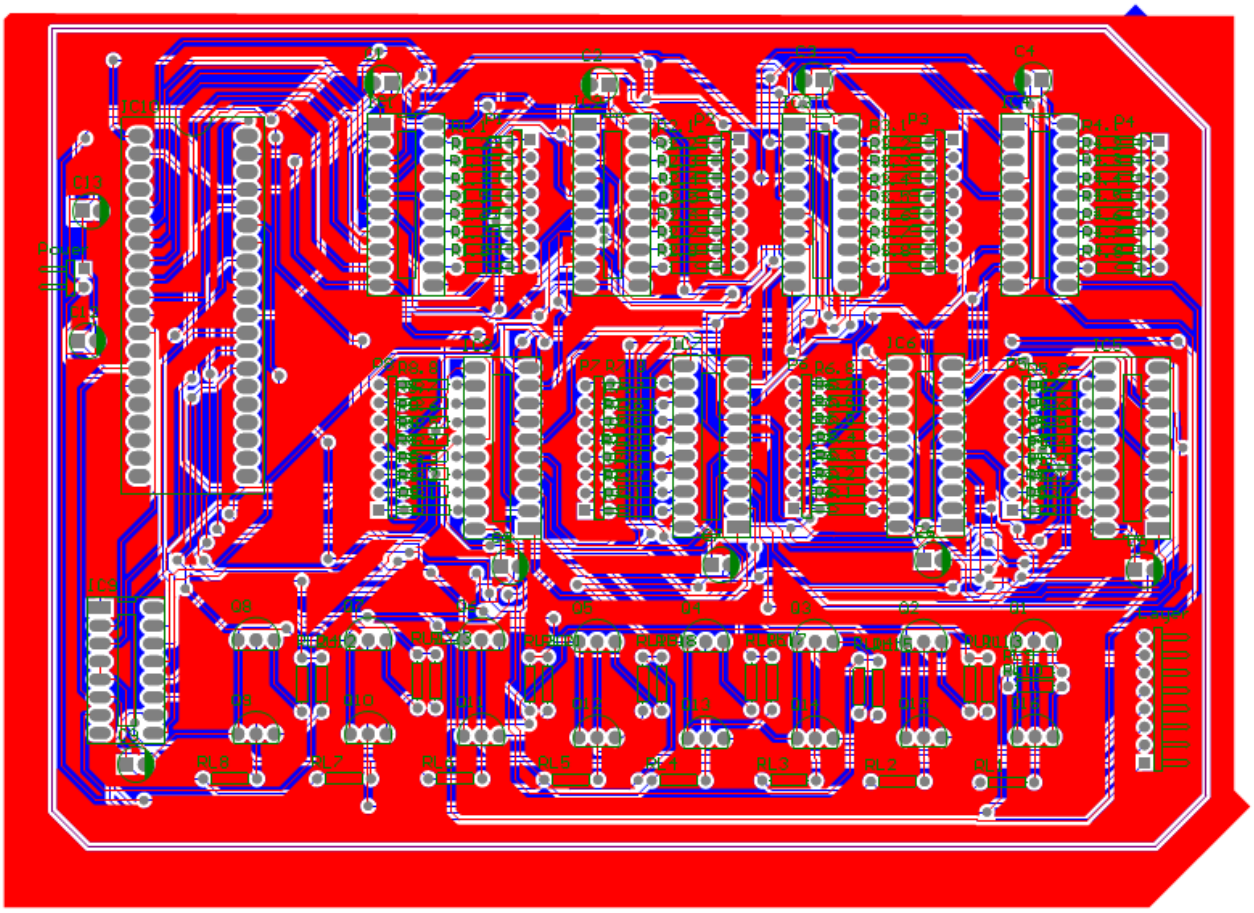


Each layer is connected to a pair of NPN transistors in parallel. The transistors are responsible for switching on and off ground for each layer in the led cube.

8.5 Box design: ratio 1:2

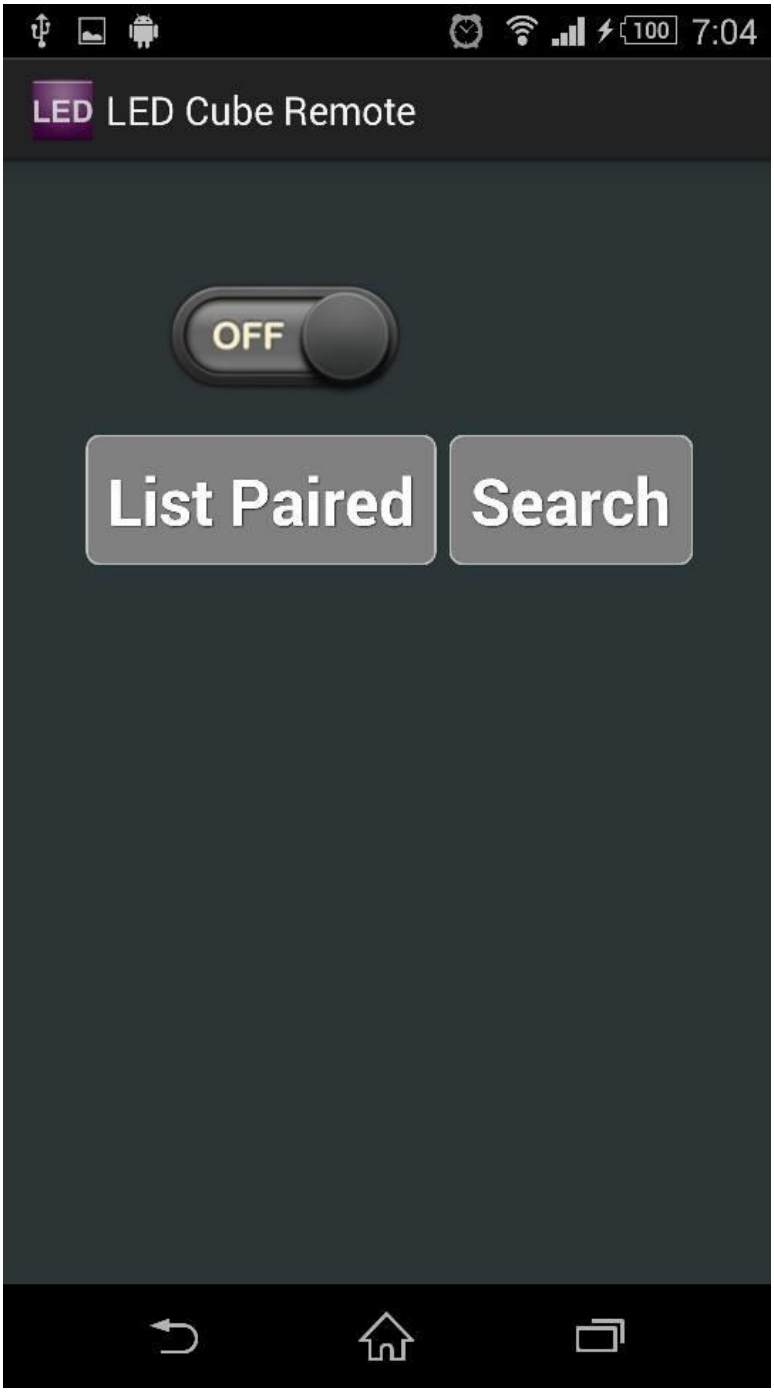


8.6 PCB design:

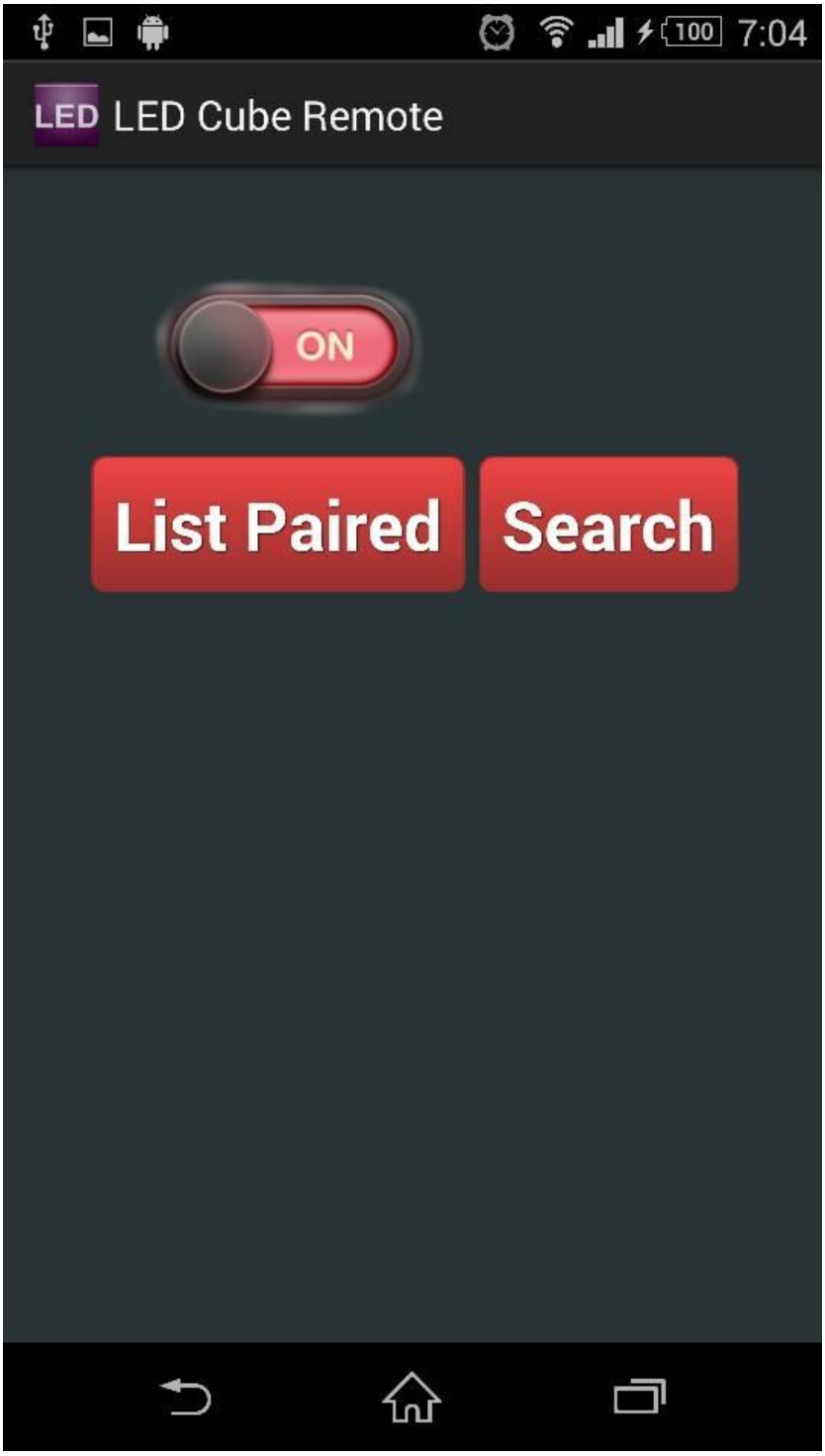


IX – Software User Interface Design

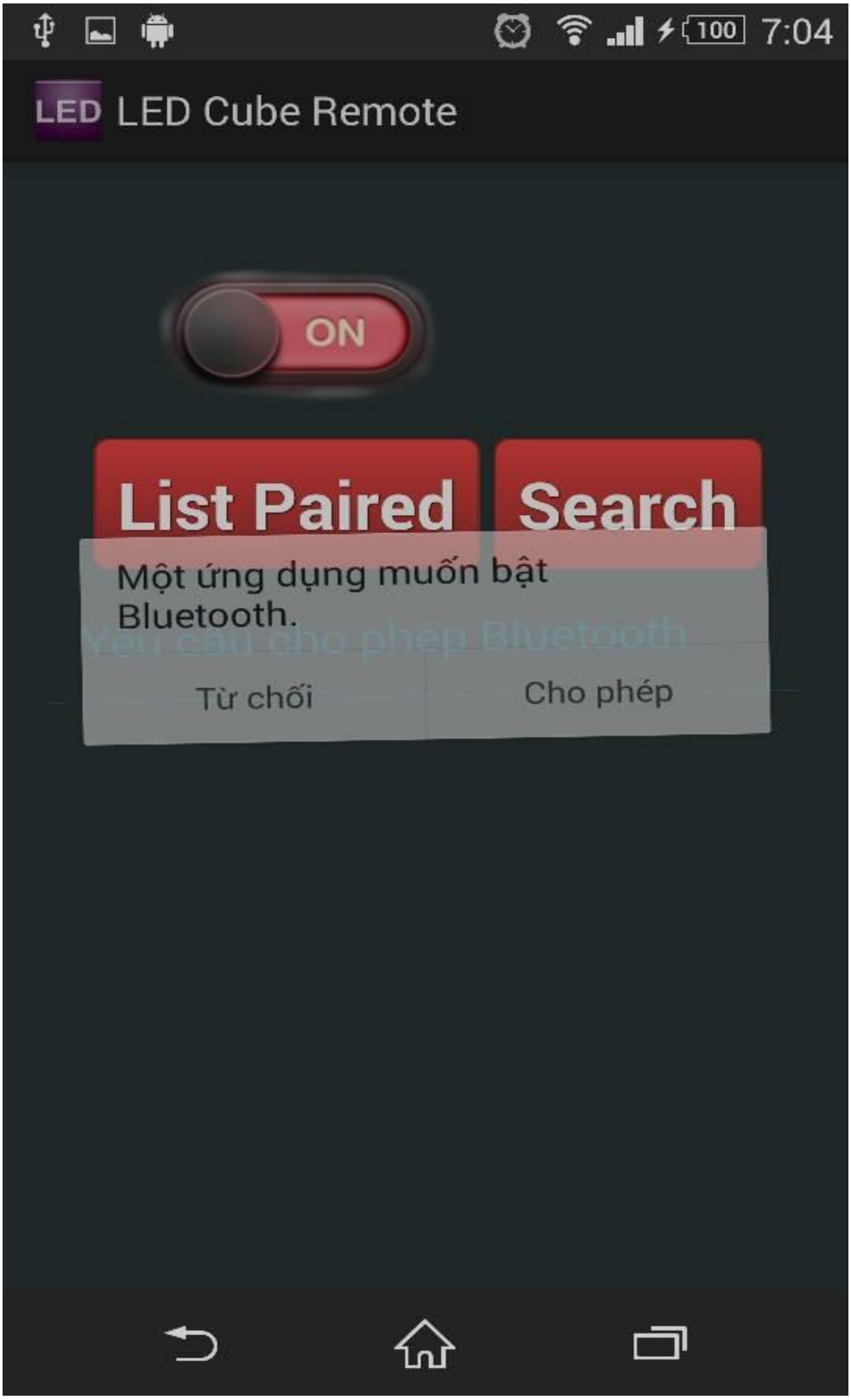
When Bluetooth is disable



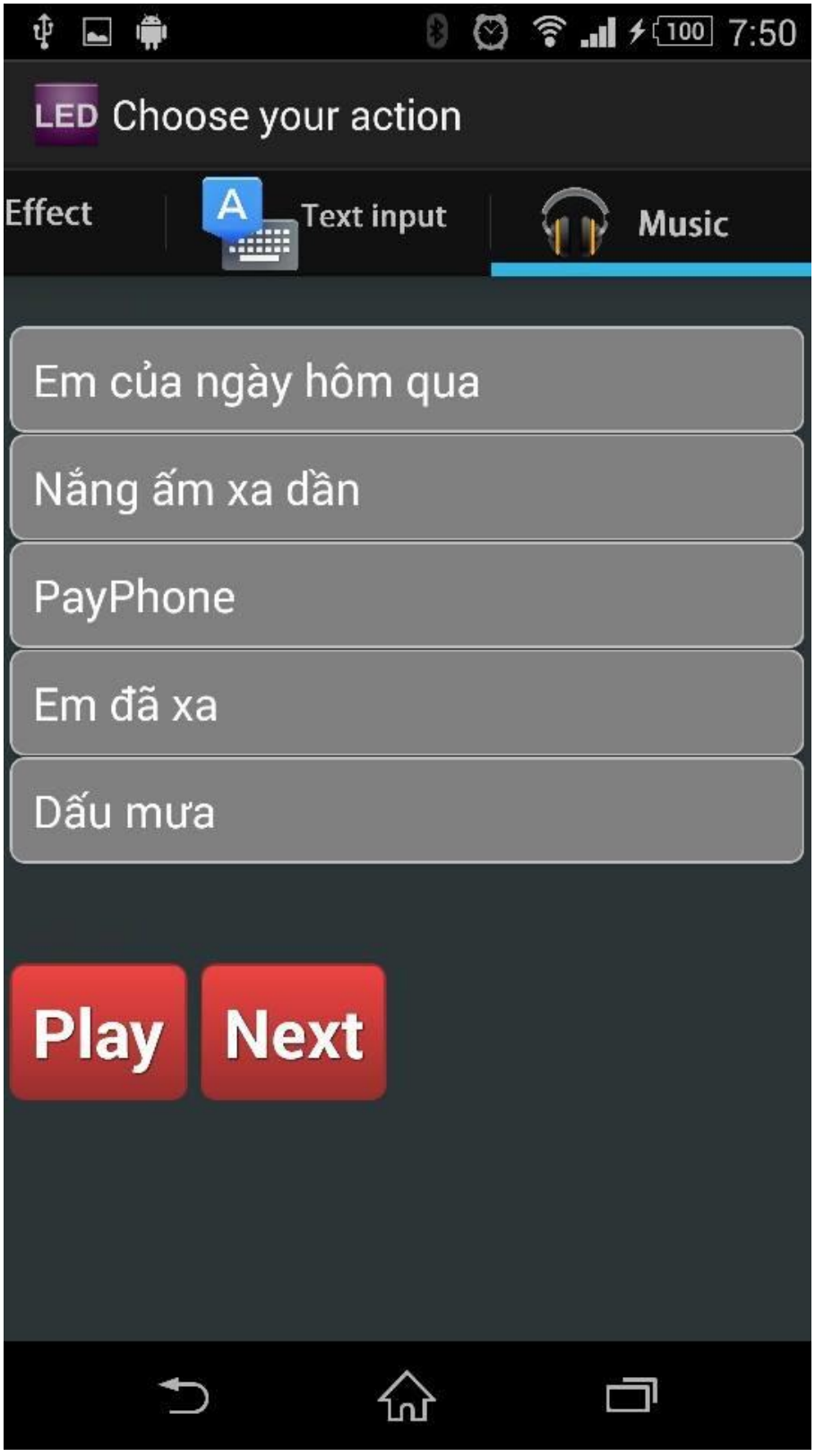
When Bluetooth is enable



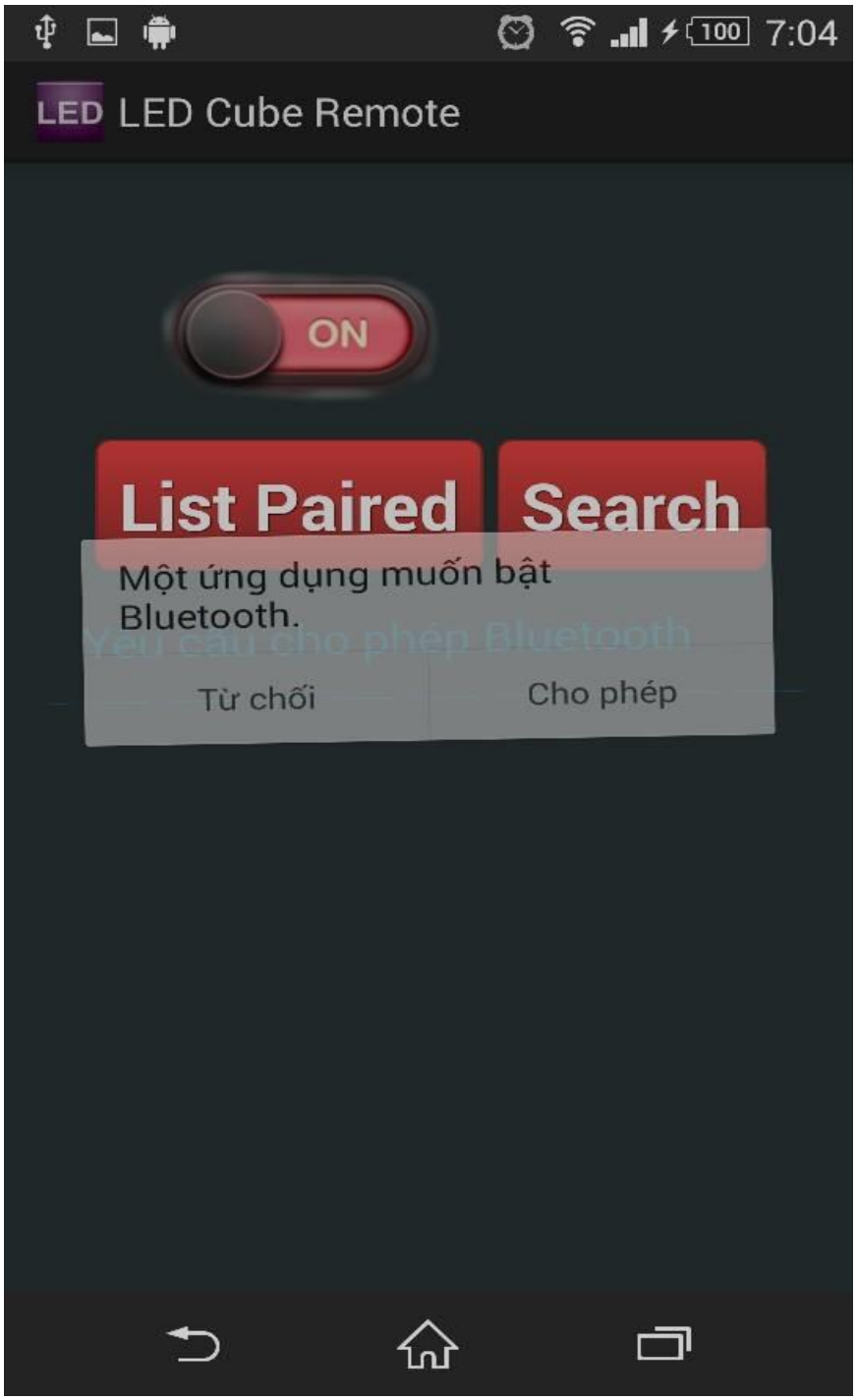
Request turn on Bluetooth



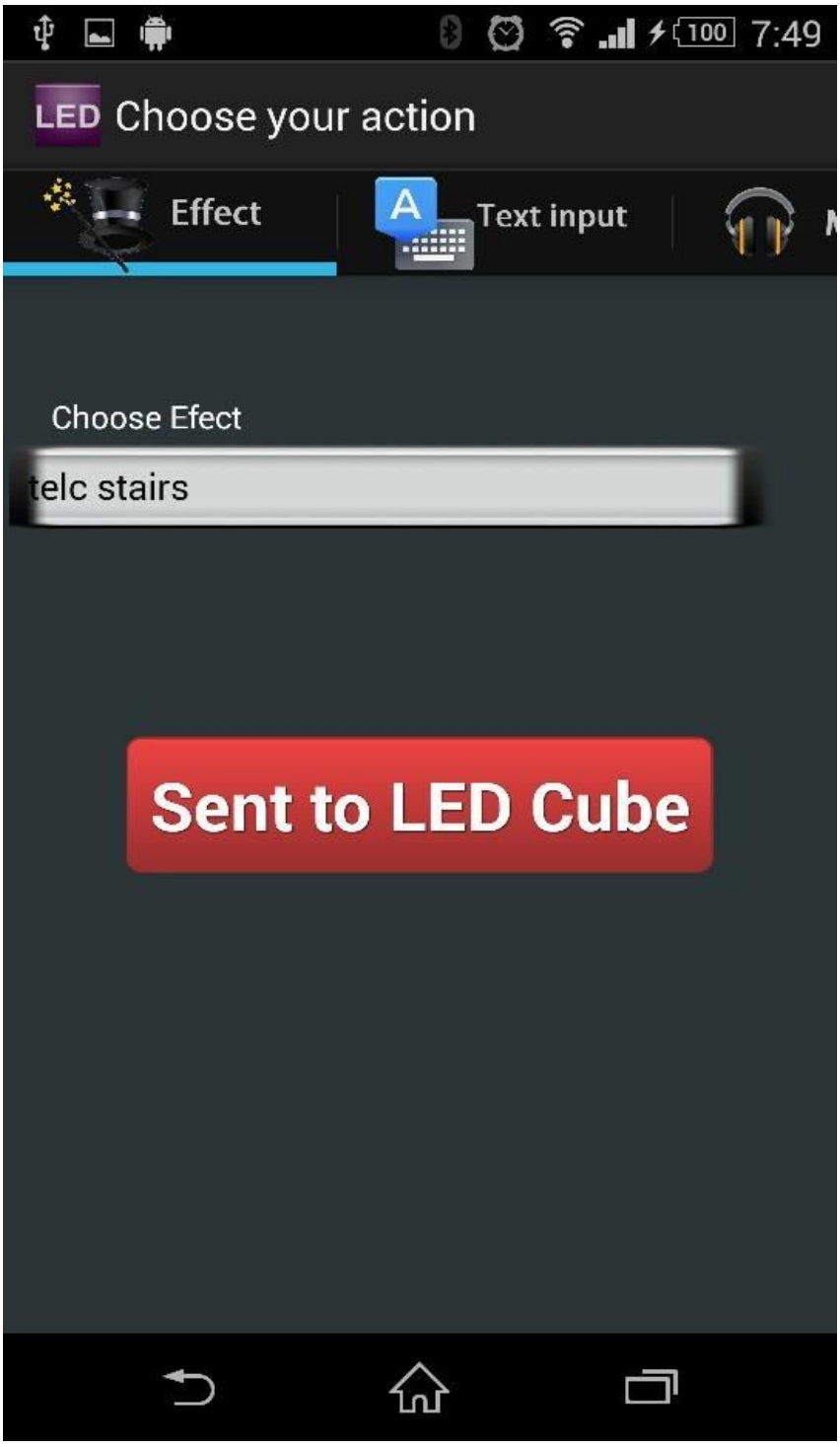
When click “List paired” button



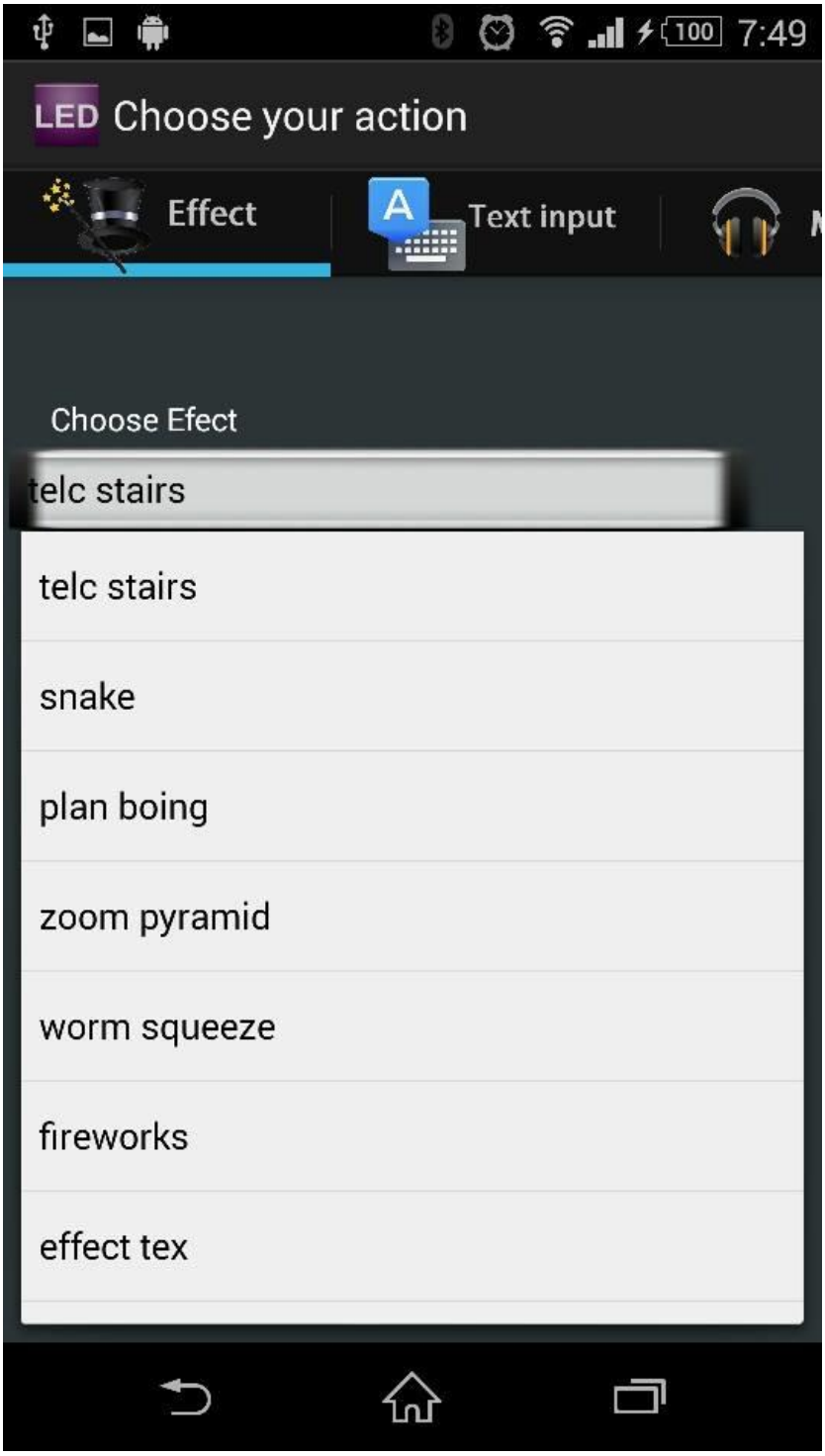
When click “Search” button



Click to Device to connect and turn to choose action screen, first stage is choose Effect

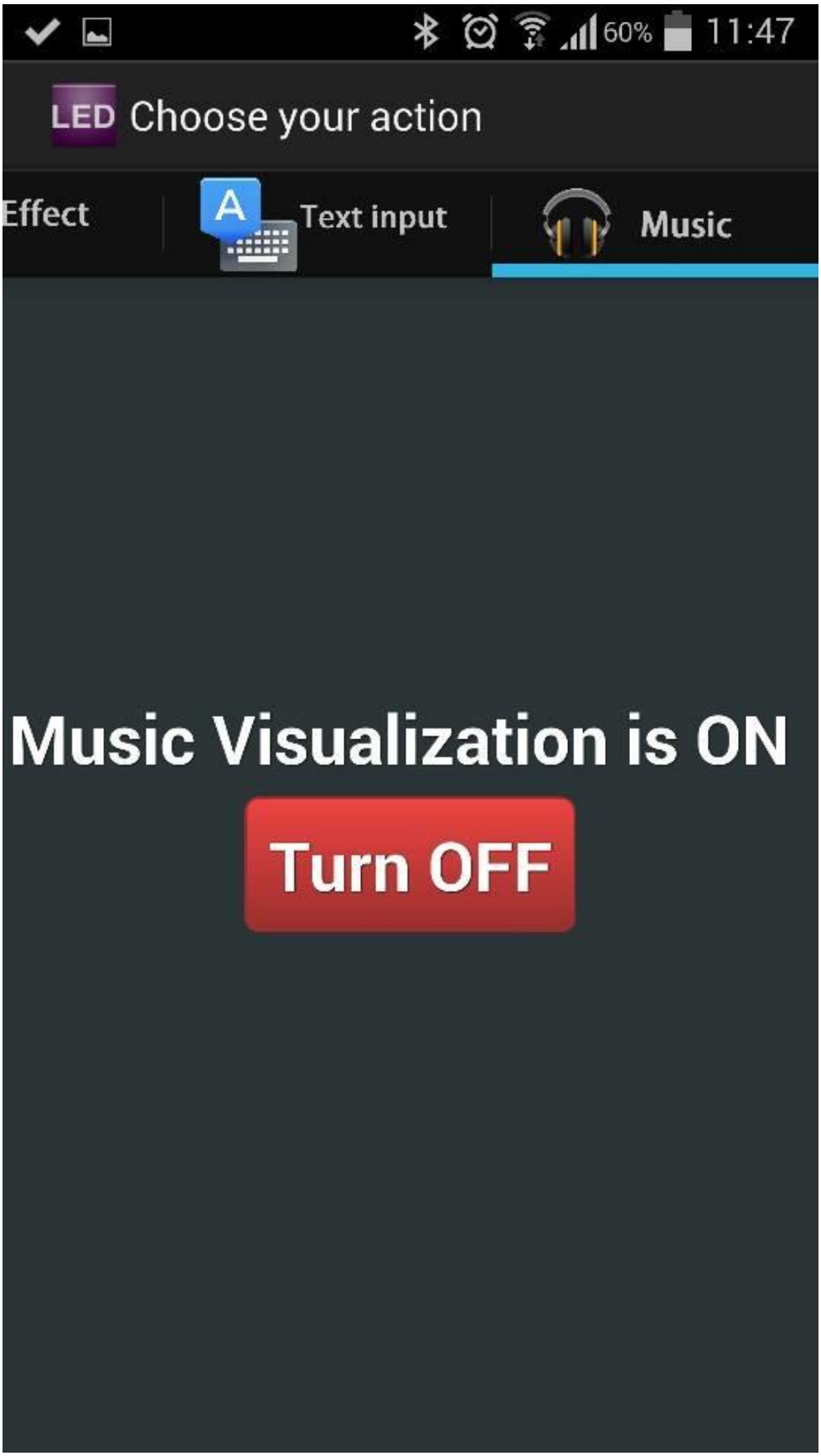


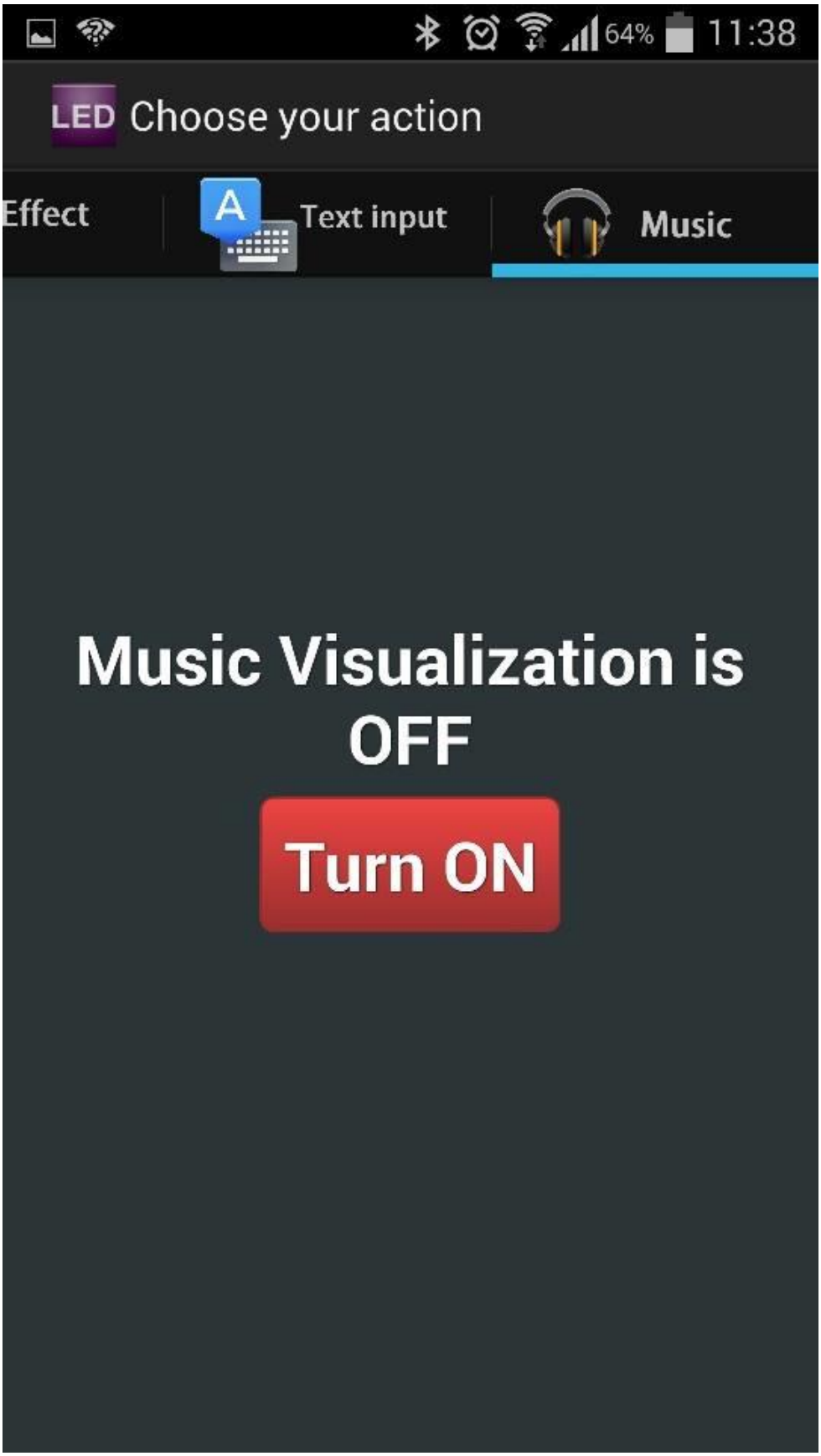
When click to Spinner which store the effects list



Swipe or click to Text tab to turn to input text screen

Swipe or click to Music tab to turn to music screen


60%
11:47
LED Choose your action
Effect
Text input
Music
Music Visualization is ON
Turn OFF


64%
11:38
LED Choose your action
Effect
Text input
Music
Music Visualization is OFF
Turn ON

X – Algorithms for LED Effects

In LCR project we realize that there are a lot of things we can do on this Cube, such as light up the room, display a message and many beautiful visualization. In limit of Arduino pro-mini kit's flash memory size, we decided to choose 13 effects: 1 for the text and 2 for the music visualization and the others for normal use. Described in a list below:

- Displaying Text: Effect text
- Music visualization:
  - Boombox
- For normal use:
  - Default Effects
  - Firework
  - Rain
  - Planboing
  - Ripples
  - Filip filop
  - Axis up-down random send parallel
  - Box shrink grow and woop woop
  - Side waves
  - TelcStair
  - Linespin

Here is how each effect works:

1. Default Effects

Default effects including all below normal use effects

2. Rain

This effect adds raindrops to the top layer of the cube, then lets them fall down to the bottom layer.

Most of the effects have a main for() loop that loops from i=0 to i < iterations. effect_rain(int iterations) only takes one argument, which is the number of iterations.

Inside the iteration loop, the function does the following:

1) Create a random number between 0 and 3, let's call it n here.

2) Loop a for() loop n number of times.

3) For each iteration of this loop, place a pixel on layer 7 (z=7) at random x and y coordinates.

4) Delay for a while

5) Shift the contents of the entire cube along the Z axis by -1 positions. This shifts everything down one level.

3. Planboing

This effect draws a plane along the specified axis then moves it from position 0 to 7 on the axis and back again. This is very simple, but it really brings out the depth of the 3d LED cube :)

This function doesn't have an iteration loop. Instead it is called twice for each axis in launch_effect().

Here is what it does:

1) For()-loop i from 0 to 7.

2) Clear the cube with fill(0x00);

3) Call setplane() to draw a plane along the desired axis at position i. The plane isn't actually drawn on the axis specified, it is drawn on the other two axis. If you specify AXIS_Z, a plane is drawn on axis X and Y. It's just easier to think of it that way. Instead of having constants named PLANE_XY, PLANE_YZ etc.

4) Delay for a while.

5) Repeat the same loop with i going from 7 to 0.

4. Ripples

The effect you see is actually just a sine wave emanating from bottom to top and vice in the center of the cube, x=3.5, y=3.5.

Here is how it works:

1) Loop through the iteration counter.

2) Loop through all 64 x and y coordinates.

3) Calculate the distance between the center of the cube and the x/y coordinate.

4) The z coordinate is calculated with sin() based on the distance from the center + the iteration counter. The result is that the sine wave moves out from the center as the iteration counter increases.

5. Axis up down random send parallel

64 voxels start on one of the side walls then shift them randomly to the opposite side of the wall

Here is how it works:

1) 64 voxels start on one of the side walls

2) Random shift one voxel one step ahead

3) Loop until that voxel reach the opposite side

4) Mean while followed by another random voxel 1 at a time

5) Loop until all voxels reach the opposite side of the wall.

6. Box shrink grow and woop woop

It takes four arguments, number of iterations, rotation, flip and delay. Rotation specifies rotation around the Z axis at 90 degree intervals. Flip > 0 flips the cube upside-down.

To make the function as simple as possible, it just draws a box from 0,0,0 to any point along the diagonal between 0,0,0 and 7,7,7 then uses axis mirror functions to rotate it.

1) Enter main iteration loop.

2) Enter a for() loop going from 0 to 15.

3) Set xyz to 7-i. This makes xyz the reverse of i. We want to shrink the box first, then grow. xyz is the point along the diagonal. We just used one variable since x, y and z are all equal along this diagonal.

4) When i = 7, the box has shrunk to a 1x1x1 box, and we can't shrink it any more. If i is greater than 7, xyz is set to i-8, which makes xyz travel from 0 to 7 when i travels from 8 to 15. We did this trick to avoid having two for loops, with one going from 7-0 and one from 0-7.

5) Blank the cube and delay a little bit to make sure the blanking is rendered on the cube. Disable the interrupt routine. We do this because the mirror functions takes a little time. Without disabling interrupts, the wireframe box would flash briefly in the original rotation before being displayed rotated.

6) Draw the wireframe box in its original rotation. side of the box is always at 0,0,0 while the other travels along the diagonal.

7) Do the rotations. If flip is greater than 0, the cube is turned upside-down. rot takes a number from 0 to 3 where 0 is 0 degrees of rotation around Z and 3 is 270 degrees. To get 270 degrees we simply mirror around X and Y.

8) Enable interrupts to display the now rotated cube.

9) Delay for a while then clear the cube.

The other function involved in the wireframe box effect is effect_box_woopwoop(). The name woopwoop just sounded natural when we first saw the effect rendered on the cube.

The woopwoop function only does one iteration and takes two arguments, delay and grow. If grow is greater than 0, the box starts as a 2x2x2 box and grow to a 8x8x8 box.

Here is how it works:

1) Clear the cube by filling the buffer with 0x00;

2) For()-loop from 0 to 3.

3) Set ii to i. If grow is specified we set it to 3-i to reverse it.

4) Draw a wireframe box centered along the diagonal between 0,0,0 and 7,7,7. One corner of the box uses the coordinates 4+ii on all axes, moving from 4-7. The other corner uses 3-ii on all axes, moving from 3-0.

5) Delay for a while, then clear the cube.

7. Filip filop

Use effect_plane_flip() to draw a plane at the top the cube then flip it by one side of that plane. Continuous flip it all over the cube.

How it works:

1) Loop through the iteration counter.

2) Start now_plane at the top

3) Determine next_plane randomly which is not as same as now_plane and opposite side of now_plane

4) Flip it using effect_plane_flip()

5) Set now_plane as next_plane.

8. TelcStair

Fill or clear the cube by a diagonal cube

How it works:

1) Loop x from 16 to 0 or vice if invert.

2) Fill smallest led cube size 1 to biggest size 8 one by one

3) Clear from smallest led cube size 1 to biggest size 8 one by one

9. Linespin

Draw a set of lines in 3d space then spin its around a pivot in center of the cube. Each line spin with different speed to make an awesome 3d animation.

How it works:

1) Loop through the iteration counter.

2) For loop z from 0 to 7.

3) Calculate top point and bottom point position in space.

4) Draw a line in space by using line_3d().

5) After drawing clear the cube by filling the buffer with 0x00.

10. Effect text – String fly

The function stringfly2 takes any ASCII string and displays it as characters flying through the cube.

It starts by placing the character at the back of the cube, then uses the shift() function to shift the cube contents towards you, making the text fly.

11. Music visualization

Boombox

Level 1:

- If the signal returns of between 114 to 128 , assign a value to a variable "i" is from 0 to 7,turn on led at positions (x=0,y=0,z=i),(x=0,y=1,z=i),(x=1,y=0,z=i),(x=1,y=1,z=i).
- If the signal returns of between 100 to 114 , assign a value to a variable "i" is from 0 to 6,turn on led at positions (x=0,y=0,z=i),(x=0,y=1,z=i),(x=1,y=0,z=i),(x=1,y=1,z=i).
- If the signal returns of between 86 to 100 , assign a value to a variable "i" is from 0 to 5,turn on led at positions (x=0,y=0,z=i),(x=0,y=1,z=i),(x=1,y=0,z=i),(x=1,y=1,z=i).
- If the signal returns of between 72 to 86 , assign a value to a variable "i" is from 0 to 4,turn on led at positions (x=0,y=0,z=i),(x=0,y=1,z=i),(x=1,y=0,z=i),(x=1,y=1,z=i).
- If the signal returns of between 58 to 72 , assign a value to a variable "i" is from 0 to 3,turn on led at positions (x=0,y=0,z=i),(x=0,y=1,z=i),(x=1,y=0,z=i),(x=1,y=1,z=i).
- If the signal returns of between 44 to 58 , assign a value to a variable "i" is from 0 to 2,turn on led at positions (x=0,y=0,z=i),(x=0,y=1,z=i),(x=1,y=0,z=i),(x=1,y=1,z=i).
- If the signal returns of between 36 to 44 , assign a value to a variable "i" is from 0 to 1,turn on led at positions (x=0,y=0,z=i),(x=0,y=1,z=i),(x=1,y=0,z=i),(x=1,y=1,z=i).
- If the signal returns of between 31 to 36 , turn on led at positions (x=0,y=0,z=0),(x=0,y=1,z=0),(x=1,y=0,z=0),(x=1,y=1,z=0).

- Delay a period of time then turn of all led of cube then delay a period of time.

Level 2:

- If the signal returns of between 240 to 256 , assign a value to a variable "i" is from 0 to 7,turn on led at positions (x=4,y=0,z=i),(x=5,y=1,z=i),(x=5,y=0,z=i),(x=4,y=1,z=i).
- If the signal returns of between 224 to 240, assign a value to a variable "i" is from 0 to 6,turn on led at positions (x=4,y=0,z=i),(x=5,y=1,z=i),(x=5,y=0,z=i),(x=4,y=1,z=i).
- If the signal returns of between 208 to 224 , assign a value to a variable "i" is from 0 to 5,turn on led at positions (x=4,y=0,z=i),(x=5,y=1,z=i),(x=5,y=0,z=i),(x=4,y=1,z=i).
- If the signal returns of between 192 to 208 , assign a value to a variable "i" is from 0 to 4,turn on led at positions (x=4,y=0,z=i),(x=5,y=1,z=i),(x=5,y=0,z=i),(x=4,y=1,z=i).
- If the signal returns of between 176 to 192 , assign a value to a variable "i" is from 0 to 3,turn on led at positions (x=4,y=0,z=i),(x=5,y=1,z=i),(x=5,y=0,z=i),(x=4,y=1,z=i).
- If the signal returns of between 160 to 176 , assign a value to a variable "i" is from 0 to 2,turn on led at positions (x=4,y=0,z=i),(x=5,y=1,z=i),(x=5,y=0,z=i),(x=4,y=1,z=i).
- If the signal returns of between 144 to 160 , assign a value to a variable "i" is from 0 to 1,turn on led at positions (x=4,y=0,z=i),(x=5,y=1,z=i),(x=5,y=0,z=i),(x=4,y=1,z=i).

- If the signal returns of between 128 to 144 , turn on led at positions (x=4,y=0,z=0),(x=5,y=1,z=0),(x=5,y=0,z=0),(x=4,y=1,z=0).
- Delay a period of time then turn of all led of cube then delay a period of time.

Level 3:

- If the signal returns of between 368 to 384 , assign a value to a variable "i" is from 0 to 7,turn on led at positions (x=2,y=2,z=i),(x=3,y=2,z=i),(x=2,y=3,z=i),(x=3,y=3,z=i).
- If the signal returns of between 352 to 368, assign a value to a variable "i" is from 0 to 6,turn on led at positions (x=2,y=2,z=i),(x=3,y=2,z=i),(x=2,y=3,z=i),(x=3,y=3,z=i).
- If the signal returns of between 336 to 352 , assign a value to a variable "i" is from 0 to 5,turn on led at positions (x=2,y=2,z=i),(x=3,y=2,z=i),(x=2,y=3,z=i),(x=3,y=3,z=i).
- If the signal returns of between 320 to 336 , assign a value to a variable "i" is from 0 to 4,turn on led at positions (x=2,y=2,z=i),(x=3,y=2,z=i),(x=2,y=3,z=i),(x=3,y=3,z=i).
- If the signal returns of between 304 to 320 , assign a value to a variable "i" is from 0 to 3,turn on led at positions (x=2,y=2,z=i),(x=3,y=2,z=i),(x=2,y=3,z=i),(x=3,y=3,z=i).
- If the signal returns of between 288 to 304 , assign a value to a variable "i" is from 0 to 2,turn on led at positions (x=2,y=2,z=i),(x=3,y=2,z=i),(x=2,y=3,z=i),(x=3,y=3,z=i).

- If the signal returns of between 272 to 288 , assign a value to a variable "i" is from 0 to 1,turn on led at positions (x=2,y=2,z=i),(x=3,y=2,z=i),(x=2,y=3,z=i),(x=3,y=3,z=i).
- If the signal returns of between 256 to 272 , turn on led at positions
- (x=2,y=2,z=0),(x=3,y=2,z=0),(x=2,y=3,z=0),(x=3,y=3,z=0).
- Delay a period of time then turn of all led of cube then delay a period of time.

Level 4:

- If the signal returns of between 492 to 512, assign a value to a variable "i" is from 0 to 7,turn on led at positions (x=6,y=2,z=i),(x=6,y=3,z=i),(x=7,y=2,z=i),(x=7,y=4,z=i).
- If the signal returns of between 476 to 492, assign a value to a variable "i" is from 0 to 6,turn on led at positions (x=6,y=2,z=i),(x=6,y=3,z=i),(x=7,y=2,z=i),(x=7,y=4,z=i).
- If the signal returns of between 460 to 476 , assign a value to a variable "i" is from 0 to 5,turn on led at positions (x=6,y=2,z=i),(x=6,y=3,z=i),(x=7,y=2,z=i),(x=7,y=4,z=i).
- If the signal returns of between 444 to 460 , assign a value to a variable "i" is from 0 to 4,turn on led at positions (x=6,y=2,z=i),(x=6,y=3,z=i),(x=7,y=2,z=i),(x=7,y=4,z=i).
- If the signal returns of between 428 to 444 , assign a value to a variable "i" is from 0 to 3,turn on led at positions (x=6,y=2,z=i),(x=6,y=3,z=i),(x=7,y=2,z=i),(x=7,y=4,z=i).

- If the signal returns of between 412 to 428 , assign a value to a variable "i" is from 0 to 2,turn on led at positions (x=6,y=2,z=i),(x=6,y=3,z=i),(x=7,y=2,z=i),(x=7,y=4,z=i).
- If the signal returns of between 396 to 412 , assign a value to a variable "i" is from 0 to 1,turn on led at positions (x=6,y=2,z=i),(x=6,y=3,z=i),(x=7,y=2,z=i),(x=7,y=4,z=i).
- If the signal returns of between 384 to 396 , turn on led at positions
- (x=6,y=2,z=0),(x=6,y=3,z=0),(x=7,y=2,z=0),(x=7,y=4,z=0).
- Delay a period of time then turn of all led of cube then delay a period of time.

Level 5:

- If the signal returns of between 624 to 640, assign a value to a variable "i" is from 0 to 7,turn on led at positions (x=0,y=4,z=i),(x=0,y=5,z=i),(x=1,y=4,z=i),(x=1,y=5,z=i).
- If the signal returns of between 608 to 624, assign a value to a variable "i" is from 0 to 6,turn on led at positions (x=0,y=4,z=i),(x=0,y=5,z=i),(x=1,y=4,z=i),(x=1,y=5,z=i).
- If the signal returns of between 592 to 608 , assign a value to a variable "i" is from 0 to 5,turn on led at positions (x=0,y=4,z=i),(x=0,y=5,z=i),(x=1,y=4,z=i),(x=1,y=5,z=i).
- If the signal returns of between 576 to 592 , assign a value to a variable "i" is from 0 to 4,turn on led at positions (x=0,y=4,z=i),(x=0,y=5,z=i),(x=1,y=4,z=i),(x=1,y=5,z=i).

- If the signal returns of between 560 to 576 , assign a value to a variable "i" is from 0 to 3,turn on led at positions (x=0,y=4,z=i),(x=0,y=5,z=i),(x=1,y=4,z=i),(x=1,y=5,z=i).
- If the signal returns of between 544 to 560 , assign a value to a variable "i" is from 0 to 2,turn on led at positions (x=0,y=4,z=i),(x=0,y=5,z=i),(x=1,y=4,z=i),(x=1,y=5,z=i).
- If the signal returns of between 528 to 544 , assign a value to a variable "i" is from 0 to 1,turn on led at positions (x=0,y=4,z=i),(x=0,y=5,z=i),(x=1,y=4,z=i),(x=1,y=5,z=i).
- If the signal returns of between 512 to 528 , turn on led at positions
- (x=0,y=4,z=0),(x=0,y=5,z=0),(x=1,y=4,z=0),(x=1,y=5,z=0).
- Delay a period of time then turn of all led of cube then delay a period of time.

Level 6:

- If the signal returns of between 752 to 768, assign a value to a variable "i" is from 0 to 7,turn on led at positions (x=4,y=4,z=i),(x=4,y=5,z=i),(x=5,y=4,z=i),(x=5,y=5,z=i).
- If the signal returns of between 736 to 752, assign a value to a variable "i" is from 0 to 6,turn on led at positions (x=4,y=4,z=i),(x=4,y=5,z=i),(x=5,y=4,z=i),(x=5,y=5,z=i).
- If the signal returns of between 720 to 736 , assign a value to a variable "i" is from 0 to 5,turn on led at positions (x=4,y=4,z=i),(x=4,y=5,z=i),(x=5,y=4,z=i),(x=5,y=5,z=i).

- If the signal returns of between 704 to 720 , assign a value to a variable "i" is from 0 to 4,turn on led at positions (x=4,y=4,z=i),(x=4,y=5,z=i),(x=5,y=4,z=i),(x=5,y=5,z=i).
- If the signal returns of between 688 to 704 , assign a value to a variable "i" is from 0 to 3,turn on led at positions (x=4,y=4,z=i),(x=4,y=5,z=i),(x=5,y=4,z=i),(x=5,y=5,z=i).
- If the signal returns of between 672 to 688 , assign a value to a variable "i" is from 0 to 2,turn on led at positions (x=4,y=4,z=i),(x=4,y=5,z=i),(x=5,y=4,z=i),(x=5,y=5,z=i).
- If the signal returns of between 656 to 672 , assign a value to a variable "i" is from 0 to 1,turn on led at positions (x=4,y=4,z=i),(x=4,y=5,z=i),(x=5,y=4,z=i),(x=5,y=5,z=i).
- If the signal returns of between 640 to 656 , turn on led at positions
- (x=4,y=4,z=0),(x=4,y=5,z=0),(x=5,y=4,z=0),(x=5,y=5,z=0).
- Delay a period of time then turn of all led of cube then delay a period of time.

Level 7:

- If the signal returns of between 880 to 896, assign a value to a variable "i" is from 0 to 7,turn on led at positions (x=2,y=6,z=i),(x=2,y=7,z=i),(x=3,y=6,z=i),(x=3,y=7,z=i).
- If the signal returns of between 864 to 880, assign a value to a variable "i" is from 0 to 6,turn on led at positions (x=2,y=6,z=i),(x=2,y=7,z=i),(x=3,y=6,z=i),(x=3,y=7,z=i).

- If the signal returns of between 848 to 864 , assign a value to a variable "i" is from 0 to 5,turn on led at positions (x=2,y=6,z=i),(x=2,y=7,z=i),(x=3,y=6,z=i),(x=3,y=7,z=i).
- If the signal returns of between 832 to 848 , assign a value to a variable "i" is from 0 to 4,turn on led at positions (x=2,y=6,z=i),(x=2,y=7,z=i),(x=3,y=6,z=i),(x=3,y=7,z=i).
- If the signal returns of between 816 to 832 , assign a value to a variable "i" is from 0 to 3,turn on led at positions (x=2,y=6,z=i),(x=2,y=7,z=i),(x=3,y=6,z=i),(x=3,y=7,z=i).
- If the signal returns of between 800 to 816 , assign a value to a variable "i" is from 0 to 2,turn on led at positions (x=2,y=6,z=i),(x=2,y=7,z=i),(x=3,y=6,z=i),(x=3,y=7,z=i).
- If the signal returns of between 784 to 800 , assign a value to a variable "i" is from 0 to 1,turn on led at positions (x=2,y=6,z=i),(x=2,y=7,z=i),(x=3,y=6,z=i),(x=3,y=7,z=i).
- If the signal returns of between 768 to 784 , turn on led at positions
- (x=2,y=6,z=0),(x=2,y=7,z=0),(x=3,y=6,z=0),(x=3,y=7,z=0).
- Delay a period of time then turn of all led of cube then delay a period of time.4

Level 8:

- If the signal returns of between 938 to 1023, assign a value to a variable "i" is from 0 to 7,turn on led at positions (x=6,y=6,z=i),(x=6,y=7,z=i),(x=7,y=6,z=i),(x=7,y=7,z=i).

- If the signal returns of between 932 to 938, assign a value to a variable "i" is from 0 to 6,turn on led at positions (x=6,y=6,z=i),(x=6,y=7,z=i),(x=7,y=6,z=i),(x=7,y=7,z=i).
- If the signal returns of between 926 to 932 , assign a value to a variable "i" is from 0 to 5,turn on led at positions (x=6,y=6,z=i),(x=6,y=7,z=i),(x=7,y=6,z=i),(x=7,y=7,z=i).
- If the signal returns of between 920 to 926 , assign a value to a variable "i" is from 0 to 4,turn on led at positions (x=6,y=6,z=i),(x=6,y=7,z=i),(x=7,y=6,z=i),(x=7,y=7,z=i).
- If the signal returns of between 914 to 920 , assign a value to a variable "i" is from 0 to 3,turn on led at positions (x=6,y=6,z=i),(x=6,y=7,z=i),(x=7,y=6,z=i),(x=7,y=7,z=i).
- If the signal returns of between 908 to 914 , assign a value to a variable "i" is from 0 to 2,turn on led at positions (x=6,y=6,z=i),(x=6,y=7,z=i),(x=7,y=6,z=i),(x=7,y=7,z=i).
- If the signal returns of between 902 to 908 , assign a value to a variable "i" is from 0 to 1,turn on led at positions (x=6,y=6,z=i),(x=6,y=7,z=i),(x=7,y=6,z=i),(x=7,y=7,z=i).
- If the signal returns of between 896 to 902 , turn on led at positions
- (x=6,y=6,z=0),(x=6,y=7,z=0),(x=7,y=6,z=0),(x=7,y=7,z=0).
- Delay a period of time then turn of all led of cube then delay a period of time.

Chapter 5 - Implementation and Testing

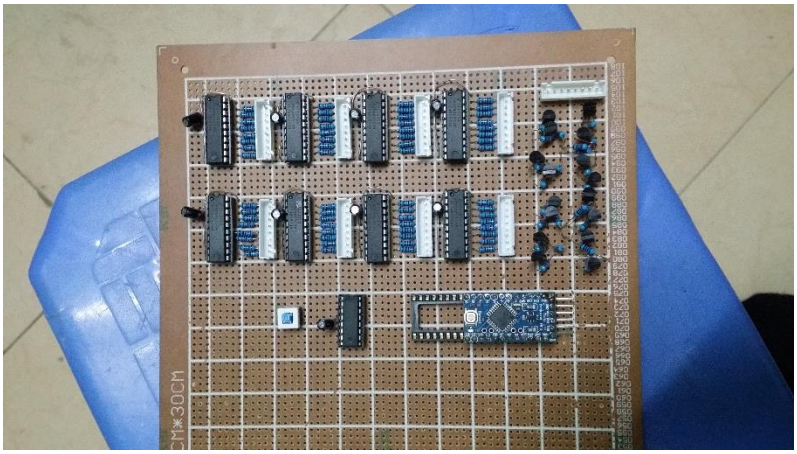
XI – Implementation

There are two phase when we implement product

11.1 Phase 1

In this phase, we create a sample product to test and research

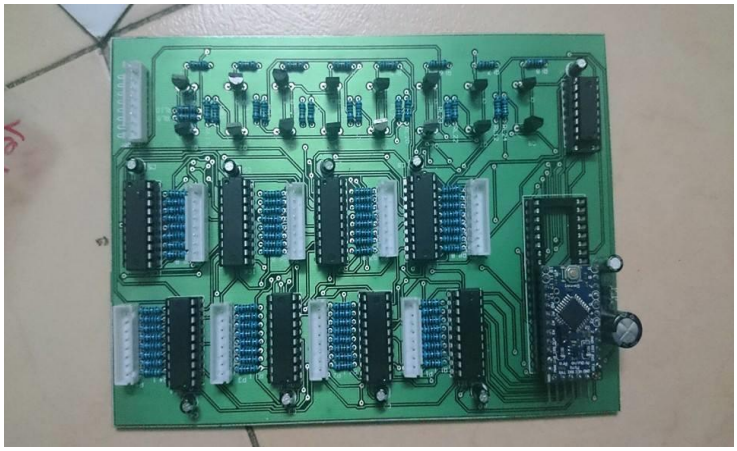
Some picture:

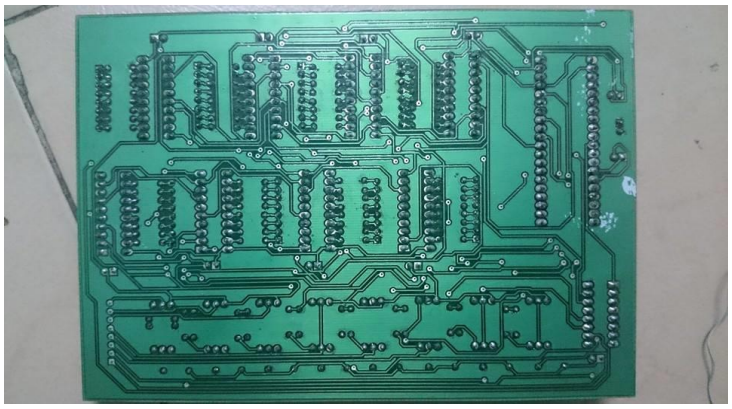


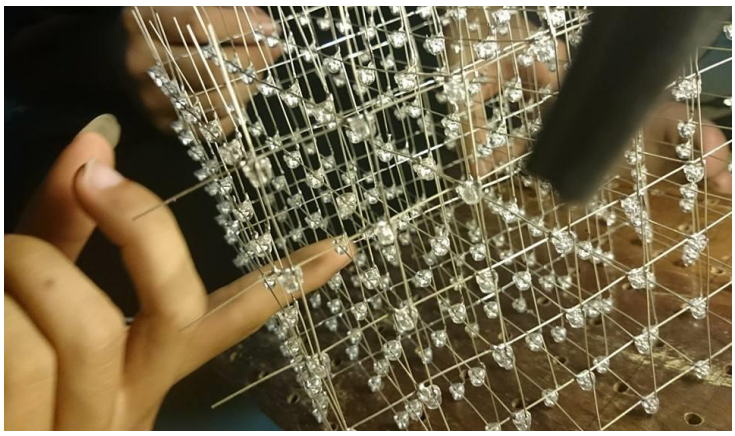
Test effect sample code

11.2 Phase 2

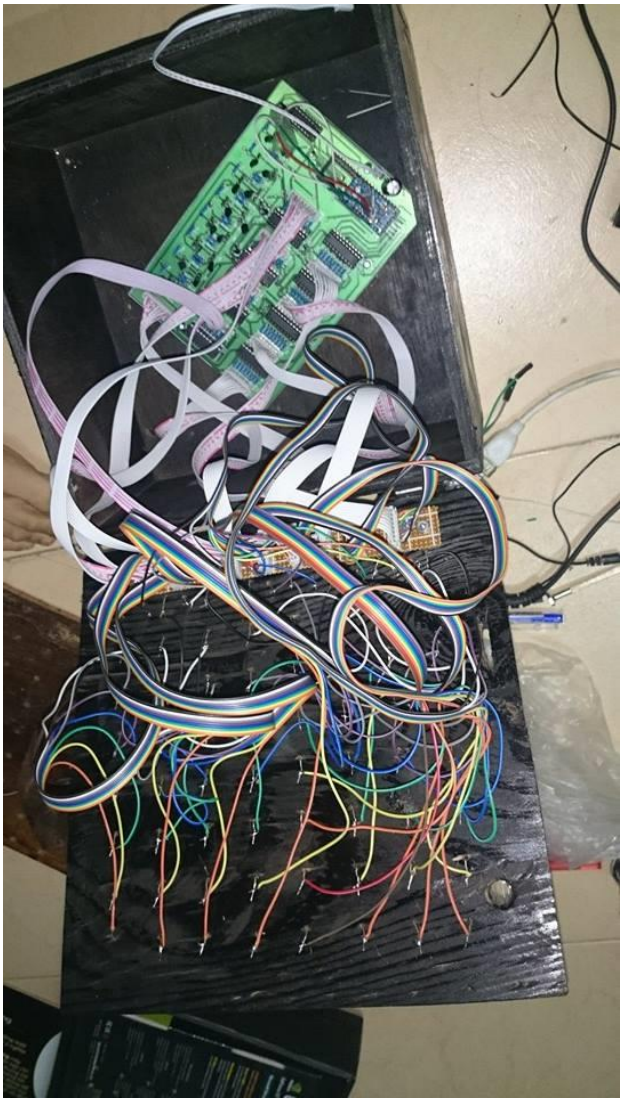
In phase 2, we try to made beauty product with new LED Cube and new printed circuit







And make a box to protect components



XII - Testing

For every project, testing is always play a very importance role in development process. Because of its role we highly concentrate on doing test in each phase of the project.

In phase 1 and phase 2 we testing on the same test case but due to different condition we still decide to take test independent in each phase of the project, the different will describes in phase 2 testing.

12.1 Test Plan

Due to implementation process separate into 2 Phases we decide to take test in each phases independent.

In phase 1 we mainly focus on test in hardware component and module in running condition to see if it run correctly. In phase 2 we mainly focus on testing in function and software protocol to see the Android Device and Led Cube communication.

Feature not to be test including:

- Age of product
- Unexpected situation like crash, falling
- The different environments condition that the LCR music sensor can working well

12.1.1 Phases test plan

	Test objective	Technique	Complete criteria	Special consideration
Phase 1	Testing hardware to ensure all the component work perfectly including led cube, IC74HC138, IC74HC547, Arduino promini kit, and control board	Tester execute this step by step to ensure everything is valid • Using an 3v power source to testing each Led and Led Cube • Using test code for Arduino to test	• All step has been execute • All identified defects have been addressed and closed	Testing may be stop when • Defect found and addressed • Require a certain test coverage

		IC74HC138 and IC74HC547 to select led and layers • Testing each effect to see if it run correctly		

Phase 2	Testing hardware to ensure all the component work perfectly including the same component on Phase 1 and Bluetooth Module HC-06 Testing LED Cube after boxing, Bluetooth Module and Sound sensor to define bug and effect running Testing the Android application to define bug and communication between Android device and LCR	Tester execute this step by step to ensure everything is valid • Using an 3v power source to testing each Led and Led Cube • Using test code for Arduino to test IC74HC138 and IC74HC547 to select led and layers • Connect module HC-06 to Arduino and sending to checking the communication Testing each effect to see if it run correctly • Running all effects including music effects by using noise and music • Using Android app to control effects and music player	• All step has been execute • All identified defects have been addressed and closed	Testing may be stop when • Defect found and addressed • Require a certain test coverage •

12.1.2 Test case form

- Phase 1 test form

Project	**LED Cube Remote**		
Author		Date	

ID	Test case	Expected result	Actual result (good/bad)	Number of test	Number of test pass	Number of test failed	Remark

- Phase 2 test form

Project	**LED Cube Remote**		
Author		**Date**	

ID	**Type of test**	**Test case**	**Expected result**	**Actual result (good/bad)**	**Number of test**	**Number of test pass**	**Number of test failed**	**Remark**
	Hardware test							
	Software test							

12.1.3 Testing tools

- 3v Power source
- Arduino IDE Serial Monitor
- Hercules version 3.2.8
- BluetoothSPP android

12.1.4 Test environment

- Microsoft Window 7,8
- Android 4.4.2

12.2 Test Item

12.2.1 Phase 1

Project	**LED Cube Remote**						
Author	TaiNA			Date	9/10/2014		
ID	Test case	Expected result	Actual result	Number of test	Number of test pass	Number of test failed	Remark
PS1-01	LED	All led are light	good	512	512	0	n/a
PS1-02	LED Cube	All led are light	bad	512	420	92	Led not light or broken (FIXED)
PS1-03	Arduino promini	Loading code ok, reset ok	good	1	1	0	n/a

PS1-04	Light layer 0	Layer 0 light	good	1	1	0	n/a
PS1-05	Light layer 1	Layer 1 light	good	1	1	0	n/a
PS1-06	Light layer 2	Layer 2 light	good	1	1	0	n/a
PS1-07	Light layer 3	Layer 3 light	good	1	1	0	n/a
PS1-08	Light layer 4	Layer 4 light	good	1	1	0	n/a
PS1-09	Light layer 5	Layer 5 light	good	1	1	0	n/a
PS1-10	Light layer 6	Layer 6 light	good	1	1	0	n/a
PS1-11	Light layer 7	Layer 7 light	good	1	1	0	n/a
PS1-12	Light AXIS_X 01	Horizontal row 1 light	good	1	1	0	n/a
PS1-13	Light AXIS_X 02	Horizontal row 2 light	good	1	1	0	n/a
PS1-14	Light AXIS_X 03	Horizontal row 3 light	bad	1	0	1	welds unbend (FIXED)
PS1-15	Light AXIS_X 04	Horizontal row 4 light	good	1	1	0	n/a
PS1-16	Light AXIS_X 05	Horizontal row 5 light	good	1	1	0	n/a

PS1-17	Light AXIS_X 06	Horizontal row 6 light	good	1	1	0	n/a
PS1-18	Light AXIS_X 07	Horizontal row 7 light	good	1	1	0	n/a
PS1-19	Light AXIS_X 08	Horizontal row 8 light	good	1	1	0	n/a
PS1-20	Light AXIS_Y 01	Vertical row 1 light	good	1	1	0	n/a
PS1-21	Light AXIS_Y 02	Vertical row 2 light	good	1	1	0	n/a
PS1-22	Light AXIS_Y 03	Vertical row 3 light	good	1	1	0	n/a
PS1-23	Light AXIS_Y 04	Vertical row 4 light	good	1	1	0	n/a
PS1-24	Light AXIS_Y 05	Vertical row 5 light	good	1	1	0	n/a
PS1-25	Light AXIS_Y 06	Vertical row 6 light	good	1	1	0	n/a
PS1-26	Light AXIS_Y 07	Vertical row 7 light	good	1	1	0	n/a
PS1-27	Light AXIS_Y 08	Vertical row 8 light	good	1	1	0	n/a

PS1-28	Fill LED Cube	All led are light	bad	1	0	1	5 led broken (FIXED)
PS1-29	Effect rain	Effect running correctly	good	1	1	0	n/a
PS1-30	Effect planboing	Effect running correctly	good	1	1	0	n/a
PS1-31	Effect ripples	Effect running correctly	good	1	1	0	n/a
PS1-32	Effect intro	Effect running correctly	good	1	1	0	n/a

12.2.2 Phase 2

Project	**LED Cube Remote**							
Author	TaiNA				Date	4/12/2014		
ID	**Type of test**	**Test case**	**Expected result**	**Actual result**	**Number of test**	**Number of test pass**	**Number of test failed**	**Remark**
PS3-01	Hardware test	LED	All led are light	good	512	512	0	n/a
PS3-02		LED Cube	All led are light	bad	512	498	14	Led not light or broken (FIXED)
PS3-03		Light layer 0	Layer 0 light	good	1	1	0	n/a
PS3-04		Light layer 1	Layer 1 light	good	1	1	0	n/a
PS3-05		Light layer 2	Layer 2 light	good	1	1	0	n/a

PS3-06		Light layer 3	Layer 3 light	good	1	1	0	n/a
PS3-07		Light layer 4	Layer 4 light	bad	1	0	1	IC74HC138 (replaced)
PS3-08		Light layer 5	Layer 5 light	good	1	1	0	n/a
PS3-09		Light layer 6	Layer 6 light	good	1	1	0	n/a
PS3-10		Light layer 7	Layer 7 light	good	1	1	0	n/a
PS3-11		Light AXIS_X 01	Horizontal row 1 light	good	1	0	1	IC74HC547 (replaced)
PS3-12		Light AXIS_X 02	Horizontal row 2 light	good	1	0	1	IC74HC548 (replaced)
PS3-13		Light AXIS_X 03	Horizontal row 3 light	bad	1	0	1	IC74HC549 (replaced)
PS3-14		Light AXIS_X 04	Horizontal row 4 light	good	1	1	0	n/a
PS3-15		Light AXIS_X 05	Horizontal row 5 light	good	1	1	0	n/a
PS3-16		Light AXIS_X 06	Horizontal row 6 light	good	1	1	0	n/a
PS3-17		Light AXIS_X 07	Horizontal row 7 light	good	1	1	0	n/a
PS3-18		Light AXIS_X 08	Horizontal row 8 light	good	1	0	1	IC74HC547 (replaced)
PS3-19		Light AXIS_Y 01	Vertical row 1 light	good	1	1	0	n/a
PS3-20		Light AXIS_Y 02	Vertical row 2 light	good	1	1	0	n/a
PS3-21		Light AXIS_Y 03	Vertical row 3 light	good	1	1	0	n/a
PS3-22		Light AXIS_Y 04	Vertical row 4 light	good	1	1	0	n/a
PS3-23		Light AXIS_Y 05	Vertical row 5 light	good	1	1	0	n/a
PS3-24		Light AXIS_Y 06	Vertical row 6 light	good	1	1	0	n/a
PS3-25		Light AXIS_Y 07	Vertical row 7 light	good	1	1	0	n/a
PS3-26		Light AXIS_Y 08	Vertical row 8 light	good	1	1	0	n/a
PS3-27		Fill led cube	Light up all led	good	1	1	0	n/a

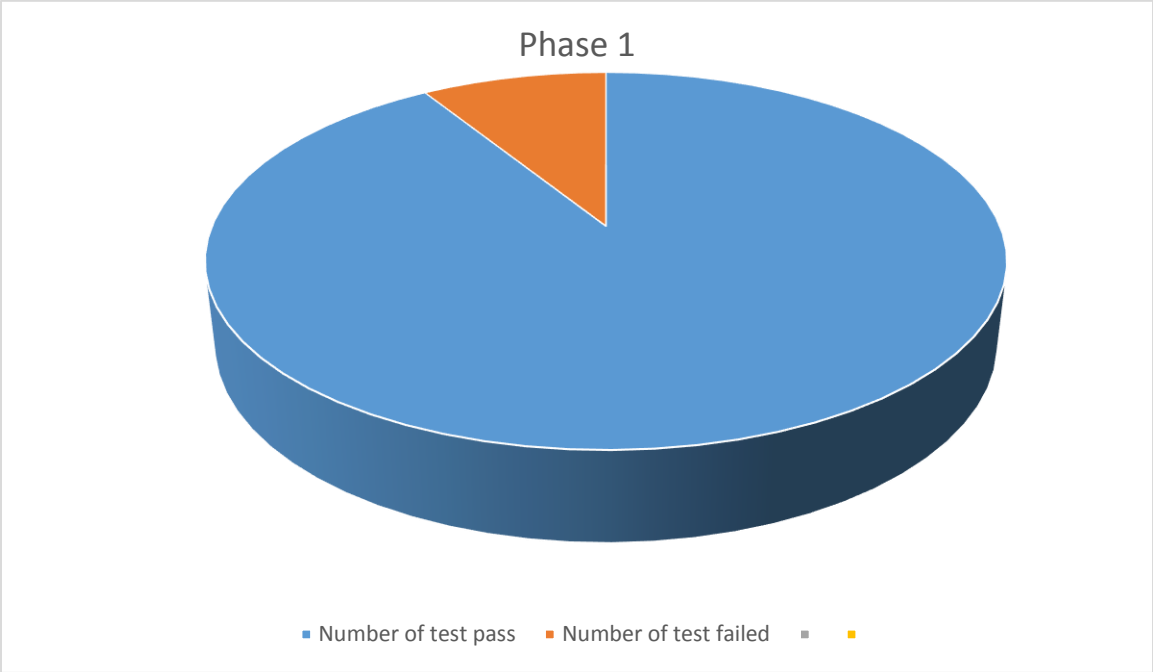
PS3-28		Default effects	Turn on the cube an all the default effects run perfectly	good	1	1	0	n/a
PS3-29		Sound sensor	Led signal blink by sound it detects	good	4	2	2	n/a
PS3-30		Sound sensor digital out	Led blink by sound it detects	good	3	3	0	n/a
PS3-31		Default music effect	The default music effect run perfectly	good	3	3	0	n/a
PS3-32	**Software testing**	Bluetooth connection	Open the Android application connect it to LC	good	1	1	0	n/a
PS3-33		Layout test	After bluetooth connect open app layout and choosing different tab	good	1	1	0	n/a
PS3-34		Select Effect - Rain	Effect rain start after chose	good	1	1	0	n/a
PS3-35		Select Effect - Rain	Effect rain start after chose	good	1	1	0	n/a
PS3-36		Select Effect - Planboing	Effect rain start after chose	good	1	1	0	n/a
PS3-37		Select Effect - Filip filop	Effect rain start after chose	good	1	1	0	n/a
PS3-38		Select Effect - Firework	Effect rain start after chose	good	1	1	0	n/a
PS3-39		Select Effect - TelcStair	Effect rain start after chose	good	1	1	0	n/a

PS3-40		Select Effect - Axis up down random send parallel	Effect rain start after chose	good	1	1	0	n/a
PS3-41		Select Effect - Box shrink grow	Effect rain start after chose	good	1	1	0	n/a
PS3-42		Select Effect - Ripples	Effect rain start after chose	good	1	1	0	n/a
PS3-43		Select Effect - Linespin	Effect rain start after chose	good	1	1	0	n/a
PS3-44		Select Effect - Sidewave	Effect rain start after chose	good	1	1	0	n/a
PS3-45		Sending text - Type on textbox	String appear in textbox	good	1	1	0	n/a
PS3-46		Sending text - press button send	String send to arduino	good	1	1	0	n/a
PS3-47		Effect text	String cut into char and appear from front to back	good	1	1	0	n/a
PS3-48		Music tab	Touch "Turn on" button - button change to "Turn off"	good	1	1	0	n/a
PS3-49		Music tab	Touch "Turn on" button - Led cube start to react to sound around	good	1	1	0	n/a

PS3-50		Music tab	Touch "Turn on" button - Text change to "Music visualization is on"	good	1	1	0	n/a
PS3-51		Music tab	Touch "Turn off" button - button change to "Turn on"	good	1	1	0	n/a
PS3-52		Music tab	Touch "Turn off" button - Led cube stop react to sound	good	1	1	0	n/a
PS3-53		Music tab	Touch "Turn off" button - text change to "Music visualization is off"	good	1	1	0	n/a
PS3-54		Music tab	Choose another effect button change to "Turn on"	good	1	1	0	n/a
PS3-55		Music tab	Choose another effect text change to "Music visualization is off"	good	1	1	0	n/a
PS3-56		Switch effect	Effect change when choose another effect	good	1	1	0	n/a

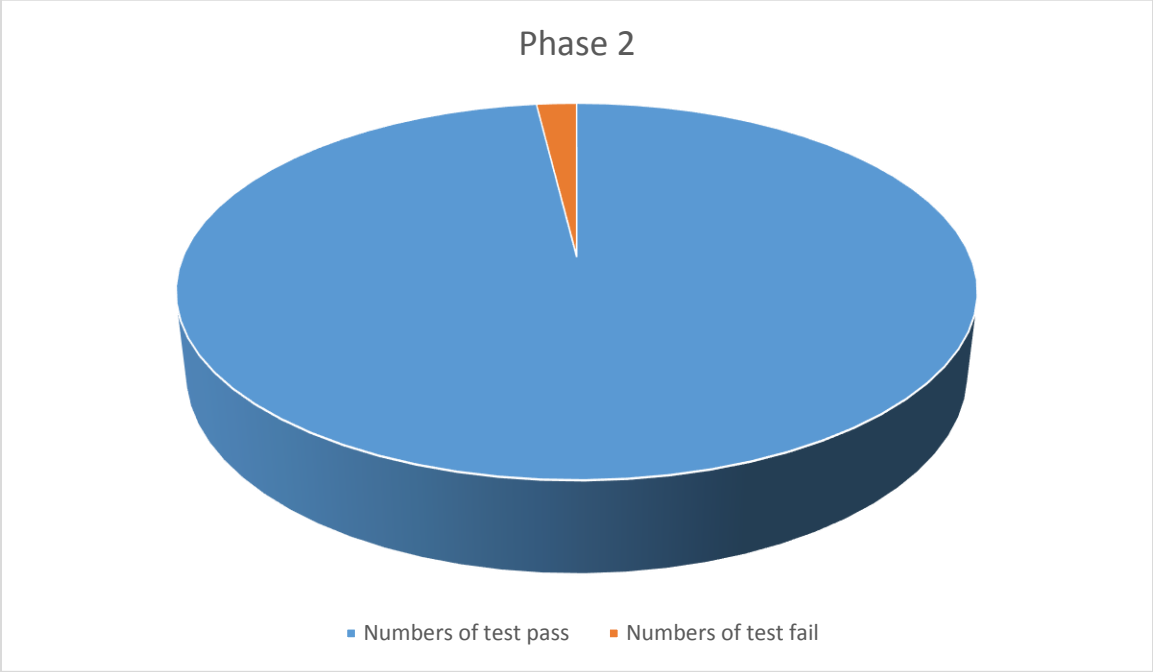
12.3 Test Report

12.3.1 Test report from Phase 1



Total number of test case	Number of test	Number of test pass	Number of test failed
32	1054	960	94

12.3.2 Test report from Phase 2



Total number of test case	Number of test	Number of test pass	Number of test failed
56	1085	1064	21

Chapter 6 - User Manual

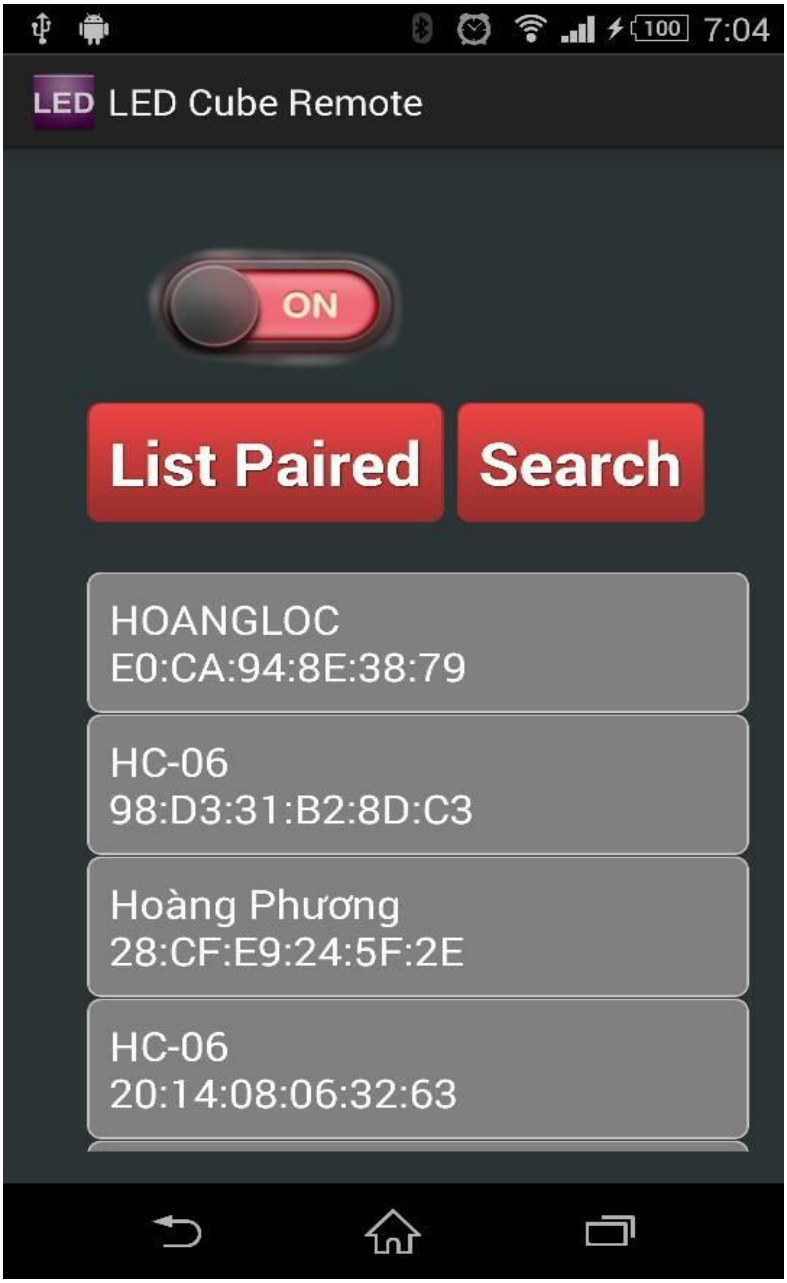
XIII - User Manual

Led Cube for activities

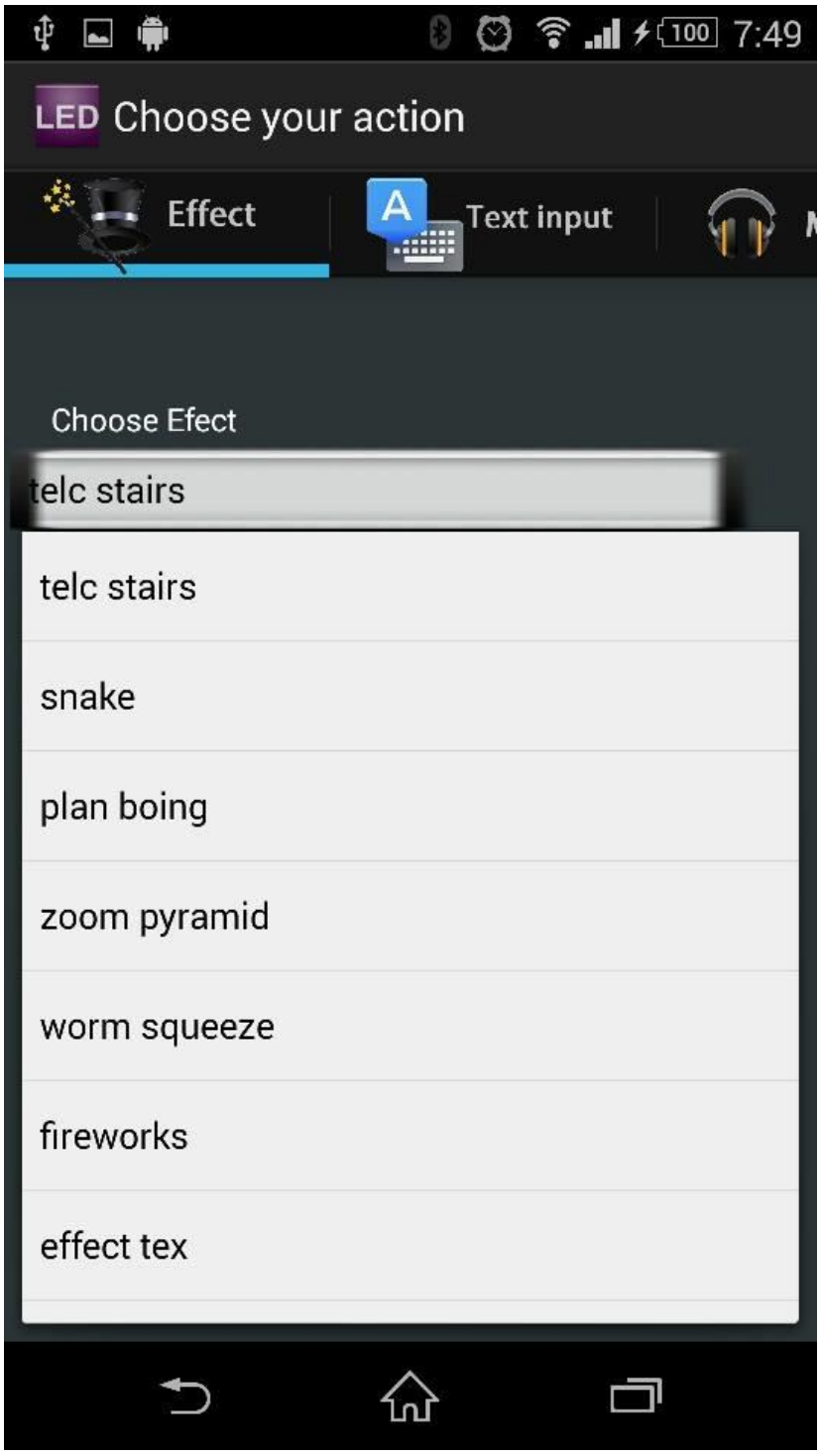
- Turn on switch to turn on the system, the system will run the default effects

- Connect Bluetooth to the control system via android app.

    + First open the app and turn on Bluetooth on your phone, then wait for the phone searching for the device.

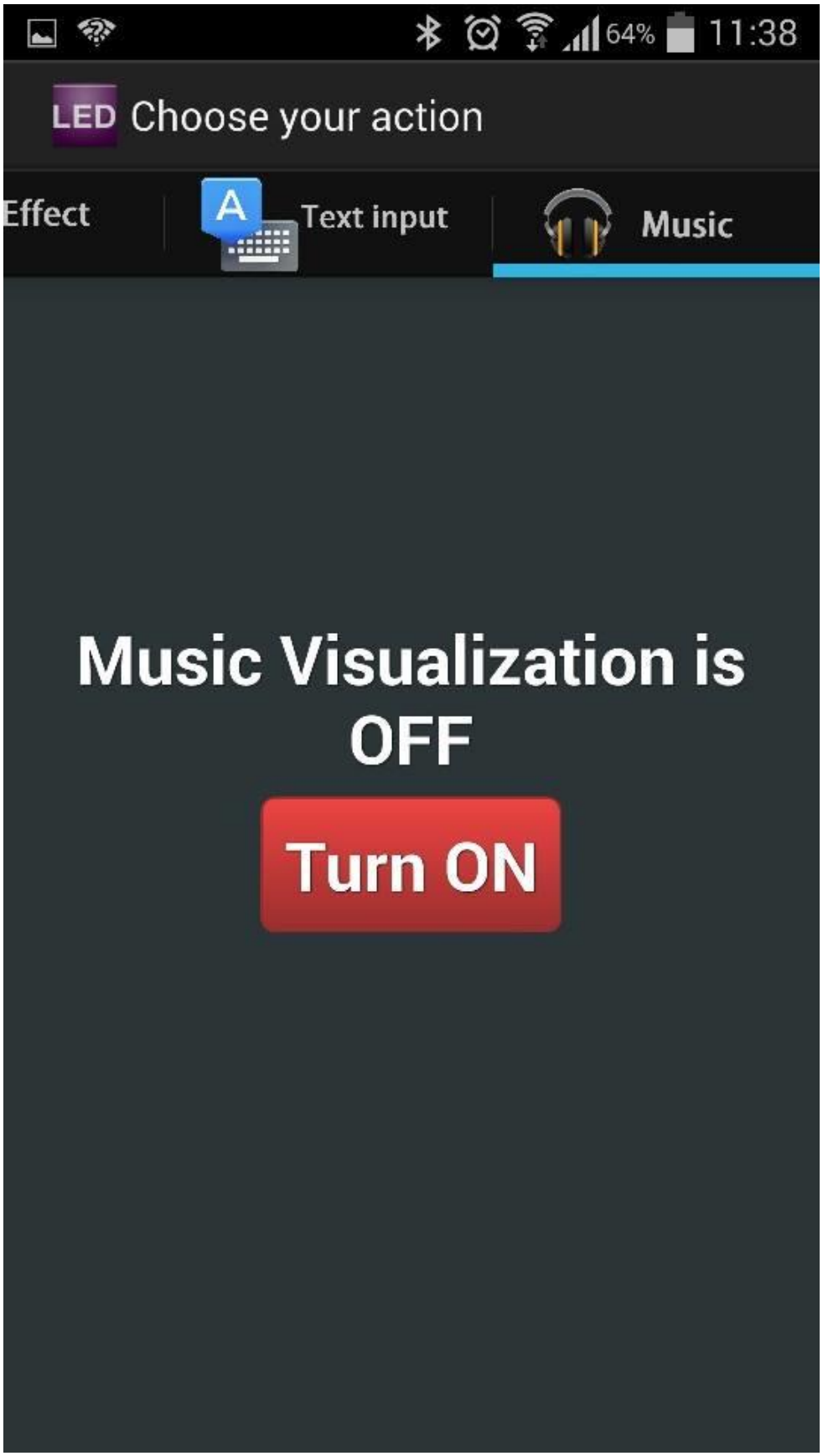


- Second select Led Cube Bluetooth device then connect.
  - Interface will open up in three key areas: EFFCTS, TEXT and MUSIC.
  - At EFFCTS page: select any effects to run this effect.



- At TEXT page: add text to the "Add Text" field and press button “Sent to LED Cube” to run Text Effect with your text.

- At MUSIC page:
  - Touch Turn-on button to turn on sound sensor and music visualization


64%
11:38
LED Choose your action
Effect
Text input
Music
Music Visualization is OFF
Turn ON

- Touch “Turn off” button to turn of music visualization

SS